'EC-0011

US Army Corps of Engineers
Topographic Engineering Center

'errain Visualization for he Portable All-Source Analysis Work Station

T

E

C

ɔral Defense Systems - Akron
210 Massillon Road
kron, OH 44315

ctober 1991

proved for public release; distribution is unlimited.

pared for:
.S. Army Corps of Engineers
)pographic Engineering Center
)rt Belvoir, Virginia 22060-5546

U.S. ARMY TOPOGRAPHIC ENGINEERING CENTER
TEC

Destroy this report when no longer needed.
Do not return it to the originator.

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

The citation in this report of trade names of commercially available products does not constitute official endorsement or approval of the use of such products.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	October 1991	Technical Report June 1990 - October 1991

4. TITLE AND SUBTITLE

Terrain Visualization for the Portable All-Source Analysis Work Station

5. FUNDING NUMBERS

DACA76-90-C-0018

6. AUTHOR(S)

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Loral Defense Systems - Akron
1210 Massillon Road
Akron, OH 44315

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

U.S. Army Topographic Engineering Center
Fort Belvoir, VA 22060-5546

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

TEC-0011

11. SUPPLEMENTARY NOTES

Effective 1 October 1991, the U.S. Army Engineer Topographic Laboratories (ETL) became the U.S. Army Topographic Engineering Center (TEC).

12a. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This report for the Terrain Visualization program was written to document significant program design elements. It reflects the successful achievement of the principal objective of the program - the implementation of sophisticated visualization capabilities using a low to medium performance computing platform. By requiring that this platform be the same as that used in the Digital Topographic Support System (DTSS), this program also advanced the potential use of terrain visualization technology in terrain analysis.

14. SUBJECT TERMS

Terrain Visualization, Terrain Analysis, Cartography, Digital Mapping

15. NUMBER OF PAGES

70

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UNLIMITED

Table of Contents

Table of Figures

Table of Tables

Preface

This report was prepared under Contract DACA76-90-C-0018 for the U.S. Army Topographic Engineering Center, Fort Belvoir, Virginia 22060-5546 by Loral Defense Systems - Akron, Akron, Ohio 44315. The Contracting Officer's Representative was John Hale.

1 Introduction

1.1 General.

This Technical Report for the Terrain Visualization program (Contract DACA76-90-C-0018) is written to document significant program design elements. It reflects the successful achievement of the principal objective of the program - the implementation of sophisticated visualization capabilities using a low to medium performance computing platform. By requiring that this platform be the same as that used in the Digital Topographic Support System (DTSS), this program also advanced the potential use of terrain visualization technology in terrain analysis.

1.2 Background.

Superior knowledge of the battlefield provides a significant decisive advantage to a field commander. Accurate visual terrain representations are of key importance in acquiring this critical battlefield information.

Formerly, terrain visualization systems have traded interactive response for image realism, or have utilized high-cost image generation hardware. The ability of terrain data to be interactively manipulated to generate accurate, realistic visual images on standard hardware would, therefore, represent a breakthrough in the capability for fielded systems to convey battlefield information.

DTSS is the U.S. Army's tool for providing automated terrain analysis capabilities to the field commander. The DTSS is currently in Full Scale Engineering Development (FSED), under the material development of the U.S. Army Engineer Topographic Laboratories (USAETL), and the program management of the Program Executive Office Command and Control Systems (PEO-CCS). Loral Defense Systems - Akron (LDS-A) is the prime contractor for the DTSS FSED. The DTSS is scheduled to achieve Milestone III in August 1992.

1.3 Objectives.

The DTSS baseline software provides the terrain analyst with the ability to generate critical Tactical Decision Aid products. The Terrain Visualization effort incorporates new technological approaches to terrain visualization, enabling enhanced transference of visual battlefield information to the field commander through previously unavailable interactive communication techniques.

The successful implementation of the Terrain Visualization software, utilizing a progressive refinement technique, offers interactive response and realistic terrain images that are implemented without high-cost image generation hardware.

The hardware and software contained in the DTSS provides the platform for the fielding of technology developed under this Advanced Concepts and Technology (ACT) initiative. The software developed under this effort is targeted for operation on the Portable All Source Analysis System Workstation (PAWS), as used in the DTSS. The successful completion of the Terrain

Visualization software development will enable rapid fielding of advanced interactive visualization software not currently funded under the DTSS program.

The utilization of a PAWS (or commercial equivalent) as a development environment enhanced the terrain visualization software development effort. A primary advantage of using the PAWS was to ease the transition of the software technology into the field environment.

1.4 Risks/Paybacks.

The foremost area of risk was in the development of interactive software for utilization on low-speed and medium-speed hardware platforms. Current systems capable of generating interactive realistic images require high-cost, high-performance image generation hardware.

The use of advanced graphics techniques and the development of low-level graphic primitives dramatically decreases system response time. Additionally, interactive response is obtained through the use of coarse initial images, that are progressively refined to increase visual realism.

The successful completion of the Terrain Visualization software enables deployment of interactive image display techniques on currently fielded systems, in addition to limiting the requirement for high cost hardware in future systems.

2 Terrain Visualization Requirements and Objectives

The following sections describe the scope of the Terrain Visualization effort.

2.1 Software Functionality.

2.1.1 Data Base Construction.

2.1.1.1 Data Base Loader.

Software was developed/modified to load Digital Terrain Elevation Data (DTED), LANDSAT data, and SPOT data into the Terrain Visualization internal data base format.

2.1.1.2 Data Base Conversion.

Software was developed/modified to convert DTSS Topographic Data Base data into the Terrain Visualization internal data base format. The contractor also was required to convert scanned map data output from a Howtek scanner into the internal Terrain Visualization data base format.

2.1.2 Interactive Parametric Input.

One of the objectives of the program was to allow the operator to be capable of interactively manipulating geometric icons to select three-dimensional parametric values defining the perspective view. The icons were to be overlaid on a reference screen which would contain a variety of background types allowing the operator to relate icon positions to real-world locations. These options included map background (Landsat, Spot, Scanned Map), a terrain sample grid, terrain-analysis source data, or continent-country outlines within a workspace boundary representing the geographic area of interest. In response to the interactive manipulation

of the icon, the software would generate a mesh perspective view within a separate graphic window. The perspective view display would be dynamically modified in response to interactive operator adjustments of the input parameters.

Once the view parameters had been finalized, the perspective-view model was to undergo progressive refinement to add realism as a function of time. The perspective view refinement stages were to consist of: Interactive Mesh Display, Hidden Line Mesh Display, Constant Shaded Display, Realistic Surface Enhancement Display, and Interim Terrain Data (ITD) feature data display.

2.1.3 Interactive Mesh Display.

The terrain was to be represented as a wire-frame mesh perspective view, representing the connection of grid posts.

2.1.4 Hidden Line Mesh Display.

Hidden lines within the perspective view, as determined by the angle of view and terrain elevation grid, would be erased from the screen to enhance the realism of the product.

2.1.5 Constant Shaded Display.

The software would generate sun-shaded representations of the terrain. The software would permit operator selection of constant shading and/or draping of the perspective view. The software would support draping of digital map data, terrain analysis products and source data, and remotely sensed imagery.

2.1.6 Realistic Surface Enhancement Display.

The software would enhance the visual realism of the gridded terrain data through operator selectable surface-enhancement techniques. The surface-enhancement techniques would include smooth shading, tensor-product-based curved surface generation, and fractal interpolation.

2.1.7 Interim Terrain Data Feature Data Overlay.

The software would support overlaying realistic representations of feature data onto the terrain. Realistic representations of feature data would include trees, drainage-lines, roads, and buildings at a minimum. These visual feature representations would be positioned on the terrain to conform with the associated coordinate data defined by the ITD. The feature size would maintain approximate scale with the terrain.

2.2 Software Environment.

The majority of the new applications were to be programmed in the Ada higher-order language as defined in ANSI/MIL-STD-1815A. In order to fully achieve interactive graphic performance, the driver routines for the graphics display were to be coded in C.

Ada is a registered trademark of the Ada Joint Program Office.

2.3 Hardware Environment

The Terrain Visualization software would be capable of operation on a PAWS, configured as utilized in the DTSS.

2.4 Development Approach.

2.4.1 Schedule.

The Terrain Visualization program was to be a 12 month effort.

2.4.2 Equipment.

As a major defense contractor, LDS-A maintains an array of computer software development systems linked over a central corporate network. The primary systems used to support the development of the Terrain Visualization software were a SUN 370 workstation and a VAXstation 2000 coupled with a MicroVAX II. These systems each offer an Ada software development environment.

As prime contractor of the DTSS, LDS-A used Government Furnished Equipment (GFE) to support the Terrain Visualization software development effort. USAETL consented to the utilization of the DTSS GFE on a non-interference basis to support the development, prototyping, and testing of the Terrain Visualization software on the target system. These GFE systems include a VAX 11/750, a MicroVAX II, and several PAWS over the life of the contract.

3 Technical Approach

3.1 Introduction.

Recent Army funded studies and contracts have advanced the automation of terrain analysis procedures and topographic support. The benefits have been revolutionary. Completely manual procedures requiring weeks of effort have been replaced by computer-based automated processes which complete in minutes. The ability of the commander to "see" the extended battlefield and make decisions based on timely, complete, and accurate terrain analysis information has become achievable. Loral Defense Systems-Akron (LDS-A) has been a leading participant in this advance.

Specific data content and format have been defined to address terrain analysis informational requirements. The definitions are collectively referred to as Tactical Terrain Data (TTD). The Defense Mapping Agency (DMA) has committed to full-scale TTD production by the mid-1990s. An Interim Terrain Data (ITD) definition currently serves to facilitate software development prior to TTD availability. ITD has been produced by DMA and distributed to development and research interests.

The Digital Topographic Support System (DTSS) will be the preeminent consumer of ITD and TTD. DTSS is a strategically transportable and tactically mobile provider of automated terrain-analysis and topographic products. DTSS is based on the PAWS ruggedized hardware platform (DEC MicroVAX II GPX and Parallax videographic processor). Terrain exploration tools developed on the PAWS platform in harmony with the DTSS development effort will easily transition to working products in the field.

The availability of digital data specific to terrain analysis interests sets apart the current environment as unique. Opportunities exist to "explore" the terrain as never before. A fundamental terrain visualization tool is the perspective view. There is no known method of communicating information to the human mind that is superior to vision. The imitation of the visual appearance of the terrain, generated through the perspective view, seeks to take advantage of this communication method.

The proposed objectives stated herein are based on ideas that seek to further terrain visualization capabilities by initially achieving a higher bandwidth of man-machine communication. In subsequent sections, a chain of ever-increasing realistic image generation algorithms are discussed. This type of approach is referred to as progressive refinement and is discussed in Section 3.2. It represents a new technological approach to terrain visualization. Each link of the chain implements an image-generation technique which improves upon the informational content of the display. The links of the progressive refinement chain are:

* **Interactive mesh display.** A graphical interface allows the operator to naturally manipulate geometrical icons to select three dimensional parametric values. A separate window contains a wire-frame display of the terrain that is updated at near-real-time rates as the orientation parameters are modified. Orientation parameter selection is discussed in Section 4.4. Interactive mesh generation is discussed in Section 4.5.

* **Hidden line mesh display.** This link in the progression removes lines which are hidden from the viewer. This represents the extent of the capabilities of most PAWS-based perspective view model predictions. Hidden-line-mesh display is discussed in Section 4.6.

* **Constant shaded display.** This element introduces surface representation to the image. Sun-shaded solid filling of surface components is implemented. Gridded data sources for the coloration, or intensity modulation, of individual polygonal elements include terrain analysis source data, remotely sensed imagery (e.g. SPOT data), and digitally scanned topographic charts. Constant shaded display is discussed in Section 4.7.

* **Realistic Surface Enhancement display.** The realistic surface enhancement link adds smooth shading, tensor-product based curved surface generation, and fractal interpolation and/or randomizing of the terrain. Realistic enhancements are discussed in Section 4.8.

* **ITD feature data overlay.** Portrayal of attribute values within the ITD format is presented. Tree, drainage-line, road, and building generations influenced by ITD attributes are included and are discussed in Section 4.9.

The above capabilities are implemented largely in the Ada language on the PAWS architecture to aid in the transition to an end product for DTSS. The following benefits were realized:

1. An improved implementation relative to the current DTSS terrain perspective view product. The new perspective view implementation will include the following capabilities to aid the transition to the DTSS program:

   A. Interactive parameter updating rates and its accompanying terrain exploration capabilities

   B. Draping of digital imagery, including scanned maps, remotely sensed data (e.g. SPOT), and terrain analysis source data

   C. Sun-shaded, solid-filled perspective views of the gridded terrain data, and draped terrain analysis source data

2. Visualization of both geometrical and non-geometrical ITD attribute information.

3. A realistic view of terrain analysis data. Attributes of the ITD data are used to realistically portray features within the data base. The USAETL's Geographic Sciences Laboratory Terrain Visualization Testbed (TVTB) currently supports this type of capability in real-time mode. This effort provides the capability to generate similar displays based on ITD data on the PAWS platform. While the CIG system performance level will not be achievable with the PAWS platform, this implementation would transition to DTSS and allow field updates to be effectively visualized.

4. Portability of the software (because of its Ada-based, system - independent nature) to other platforms which exist or become available.

These paybacks represented an opportunity, which required that certain technical risks be overcome. The foremost of these was concern for the relative performance level and graphics resources of the PAWS platform. The machine is rated at slightly less than 1 million instructions per second (MIPS). While the Parallax videographic processor is relatively fast, it is without Z-buffering and polygonal shading capabilities and allows only 8-bits per pixel in standard graphics mode. To counter this concern, experience was available in performance-oriented programming of the PAWS system on a graphics-intensive application.

3.2 Progressive Refinement.

As in other synthesis fields, two conflicting goals exist - visual realism and interactivity. Solutions which achieve both of these goals are beyond the financial resources of most programs. Consequently, much of what is central to one of the goals is often sacrificed to serve the other.

One approach to accommodating these competing demands is offered by progressive refinement. In this approach, a simple and quick version of an image is initially generated and progresses through a sequence of increasing realism until a change in the scene or viewer orientation requires that the process be restarted. The goal was to produce the highest quality image possible within time constraints demanded by operator interaction. Initial images are generated quickly and progressive details added in a graceful fashion - an automatic, continuous, non-distracting update.

The above discussion suggests the application of progressive refinement to terrain visualization. A hierarchy of terrain rendering algorithms is implemented which continually refines the quality of the displayed terrain. In addition, an interactive orientation specification method was developed, which allows the operator to select orientation rapidly and naturally. This new approach greatly improves the standard terrain perspective view capabilities expected from lower-performance computing platforms.

3.3 System Architecture And Overview.

Figure 1, which follows, shows the major functional components of the software structure. Central to the system is the on-line data base. The on-line data base consists of layers of gridded data. Each layer represents a particular informational theme (terrain, remote imagery, gridded topographic source data, etc.) and may vary in inter-sample spacing according to data type. Within individual layers, inter-sample spacing remains constant and is maintained in meters.

Functional elements shown below the on-line data base within Figure 1 are involved with data base population. Each element converts a different type of data to the on-line data base format. The system may be enhanced to support other data types with the creation of a specific conversion procedure. The Data Base Construction Parametric Input element serves to interface with the analyst for conversion-related parameter specifications.

Functional elements shown above the on-line data base within Figure 1 are involved with image generation. The progressive refinement chain is implemented within the five horizontally-aligned generation elements. All generation elements extract data from the on-line data base. The ITD Overlay Generation element also extracts data directly from the workspace GIS data base. Concurrently executing with the progressive refinement chain is the Interactive Parametric Input element. Operator-specified parametric inputs are communicated with generation elements through a common data area. Synchronization of the Interactive Parametric Input with the progressive refinement chain is accomplished through the use of Ada tasking. The General Parametric Input element interfaces with the analyst for the specification of parameters which control the progressive refinement process.

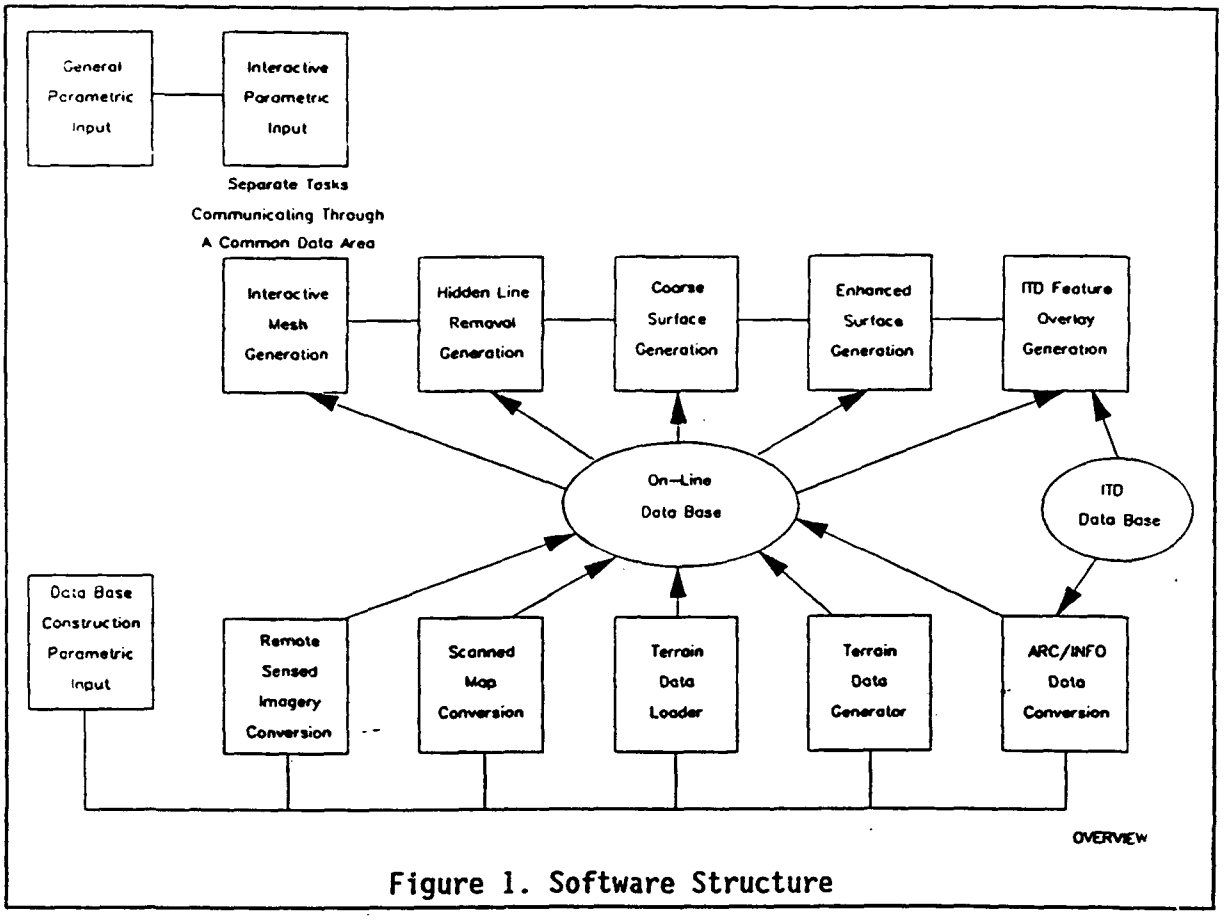


Figure 1. Software Structure

4 System Description

4.1 General Utility Function Enhancements

A math package developed for simple trigonometric functions to aid in software development and to make future implementations easier. It was noted during development that Ada does not provide a standard for math functions. As an example, for one compiler, the "atan" (inverse tangent) function required only one input (tangent, assuming radians for output), whereas in another, it required two (tangent and argument denoting mode-- radians or degrees). When a need for a function was defined, this function was added to the math library to isolate any possible future compiler differences.

In order to increase the efficiency of the access to the Parallax board, a graphics library was developed. As stated previously, this code was written in C and an Ada interface package was developed to access these routines. Routines contained within this package include: 1.) display initialization, 2.) look-up table manipulation, 3.) point draw, 4.) line draw, 5.) image draw, and 6.) image read. Direct interface via this graphics library rather than indirectly through the Graphics Kernel System (GKS) library, provided with the PAWS, accelerated graphics throughput and reduced code size. Only the routines necessary to complete this contract were included in this package.

Note that in addition to completing this library, large portions of the DTSS software would have to be rewritten in order to directly access the Parallax in the same manner. This task would require on the order of one man year in order to implement.

The graphics library was developed to use the driver software used by the DTSS instead of standard driver. This was done for 2 reasons: 1.) easier integration into DTSS systems, and 2.) avoiding the need to reboot the system to install an alternate driver during development.

4.2 Data Base Construction

The database construction function will be described in the following sections. First, there is a description of the various file formats developed for use by the Terrain Visualization software, then there is a description of the actual processing of the numerous data types.

4.2.1 File formats

4.2.1.1 Unrectified Scanned Map

The following describes the format for the unrectified scanned map images. This file format is loosely based on the Sun rasterfile format. A note on this format: although it is not necessary to have the file oriented in a particular way, it is much faster to extract the data into the tiled data format if each image line is aligned along (or nearly along) a line of longitude.

Record	Length	Contents
Header	(total 32 bytes)	
	4 bytes	magic_number (unused)
	4 bytes	image width
	4 bytes	image height
	4 bytes	image depth (bit planes)
	4 bytes	length (bytes)
	4 bytes	file_type (unused)
	4 bytes	color_map_type (unused)
	4 bytes	map_length (bytes)
Colormaps	(Total Header.map_length)	
	Header.map_length/3	Red Colormap
	Header.map_length/3	Green Colormap
	Header.map_length/3	Blue Colormap
Image Data	(Total Header.length)	
	Header.width	Line 1

	Header.width	Line 2
	...	...
	Header.width	Line Header.height

Table 1. Unrectified Scanned Map Format

4.2.1.2 SPOT Data Base Description

The SPOT data base will be a tiled database with each geounit in a separate directory. The number of data files in a directory depends upon the imagery resolution, see Table 2. Note that in Table 2, 5 meter spacing is included for the purpose of making the relationship between spacing and file size clear. Spot data currently does not support 5 meter resolution.

Spacing (meters)	Max # of files	Size of files (degrees)
20	1	1° x 1°
10	4	½° x ½°
5	16	¼° x ¼°

Table 2. SPOT Data Base Sizing

The directories are named according to data type (PAN_SPOT, ADRG, etc.) and by the southwest corner geographic degrees. For example: PAN_SPOT/n34w087.

The files are numbered with a latitude index and then a longitude index. See Figure 2.

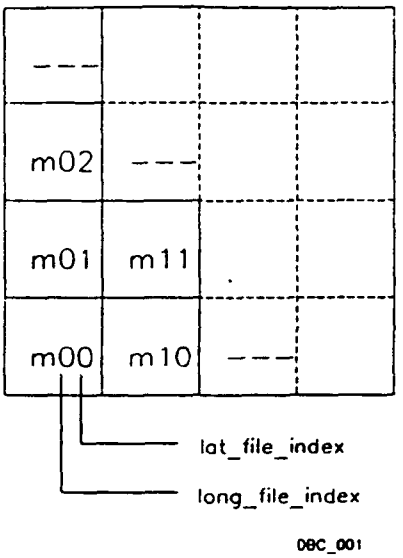


Figure 2. SPOT Data Base File Numbering

In addition to the 1, 4, or 16 matrix files, there needs to be an information file, which gives the number of rows of tiles, and the number of column of tiles (called *IY* and *IX* respectively) and a pair of integer numbers which, when divided into the number 1,296,000 (the number of arc-seconds in 360 degrees), gives the image pixel spacings in the latitude and longitude directions in arc-seconds. These numbers will be referred to as ***mlat*** and ***mlong*** (the same concept is used to avoid the use of floating point numbers for spacing in DMA's ADRG product). Also necessary is an indicator (called ***rank***) which shows the file size; 1 / rank is the coverage of a file in degrees (for example 4 indicates $\frac{1}{4}^{\circ}$.

See Table 4 for the ordering and type of values contained within the header.

rank	32-bit Integer
IX	32-bit Integer
IY	32-bit Integer
mlong	64-bit Integer
mlat	64-bit Integer

Table 3. SPOT Data Base Header File **Format**

The data base will conform to the ADRG standards, which means that the latitude and longitude pixel spacings will remain constant over given ranges of latitude, even though this results in slight variations in the distance (meter) spacings, especially in the longitude direction. Table

4 shows these standards applied to the cases of 20, 10 and 5 meters nominal spacing of the source imagery. To apply to other nominal spacings:

$$mlat = (value\ for\ 10m\ spacing) * \frac{(10.0\ meter)}{(desired\ nominal\ spacing)}$$

$$mlong = (value\ for\ 10m\ spacing) * \frac{(10.0\ meter)}{(desired\ nominal\ spacing)}$$

Latitude Range°	Nominal Spacing (m)	mlat	mlon	Lat spacing (sec)	Lon spacing (sec)
0-32	20	2001920	1848320	0.6473785	0.7011772
	10	4003840	3696640	0.3236892	0.3505886
	5	8007680	7393280	0.1618446	0.1752943
32-48	20	2001920	1512960	0.8565989	0.8565989
	10	4003840	3025920	0.4282994	0.4282994
	5	8007680	6051840	0.2141497	0.2141497
48-56	20	2001920	1228800	1.0546875	1.0546875
	10	4003840	2457600	0.5273437	0.52734375
	5	8007680	4915200	0.2636718	0.2636718
56-64	20	2001920	995840	1.3014139	1.3014139
	10	4003840	1991680	0.6507069	0.6507069
	5	8007680	3983360	0.3253534	0.3253534
64-68	20	2001920	816640	1.5869906	1.5869906
	10	4003840	1633280	0.7934953	0.7934953
	5	8007680	3266560	0.3967476	0.3967476
68-72	20	2001920	686080	1.8889925	1.8889925
	10	4003840	1372160	0.9444962	.09444962
	5	8007680	2744320	0.4722481	0.4722481
72-76	20	2001920	550400	2.3546512	2.3546512
	10	4003840	1100800	1.1773256	1.1773256
	5	8007680	2201600	0.5886627	0.5886627
76-80	20	2001920	412160	3.1444099	3.1444099
	10	4003840	824320	1.572205	1.572205
	5	8007680	1648640	0.7861024	0.7861024

Table 4. SPOT Data Base Header File Contents

Each file begins with its SW origin at an integer, half, fourth or eighth- degree point. Tiles within a file are 128 x 128 datapoints and are numbered in a manner analogous to the filename numbering system. Data within a tile progresses eastward by columns with the datapoints within a column progressing from south to north.

The tiles may extend beyond the nominal east and north file boundaries, but the datapoint values in portion of the tile which lie outside the file's boundary limits may be set to zero. See Figure 3.

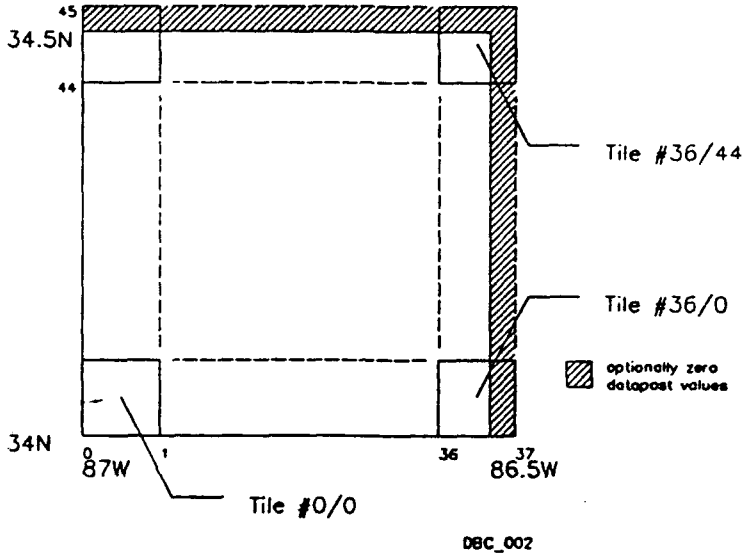


Figure 3. SPOT Tile Numbering

4.2.1.3 DTED Data Base Description

The Digital Terrain Elevation Data base will be a tiled database with each geounit in a separate directory. Both DTED I and DTED II formats are supported.

The number of data files in a directory depends upon the elevation spacing. Each file consists of 300 x 300 points. The data following shows the file coverage as a function of latitude for DTED I and DTED II.

Latitude Coverage	DTED I		DTED II	
	Cell Size	File Coverage	Cell Size	File Coverage
0 - 32	3" x 3"	15' x 15'	1" x 1"	5' x 5'
32 - 48	3" x 4"	15' x 20'	1" x 1 1/3"	5' x 6 2/3'
48 - 56	3" x 4"	15' x 20'	1" x 2"	5' x 6 2/3'
56 - 64	3" x 6"	15' x 30'	1" x 2"	5' x 10'
64 - 68	3" x 8"	15' x 40'	1" x 2 2/3"	5' x 10'
68 - 72	3" x 8"	15' x 40'	1" x 2 2/3"	5' x 13 1/3'
72 - 76	3" x 8"	15' x 40'	1" x 2 2/3"	5' x 13 1/3'
76 - 80	3" x 8"	15' x 40'	1" x 2 2/3"	5' x 13 1/3'

At the zone boundaries where the east-west spacing changes there is an additional row of files extending poleward and having the same spacing as on teh equatorward side of the boundary (i.e., at +/- 32 degrees, +/- 56 degrees and +/- 68 degrees).

There is no tiling within the files. The data runs south to north and then west to east within the files.

4.2.1.4 ITD and DTSS Product Data Formats

In order to provide data supporting the generation of models and DTSS Product data overlays within the Terrain Visualization software, ITD data and DTSS products must be transformed into a format compatible with the software. This process relies on existing DTSS ITD loader and product generation capabilities to initially generate ITD and product data files.

The conversion of DTSS data is a two step process: (1) Data from an existing DTSS directory is selected and placed into an intermediate file, (2) Further selections are applied to this file and data is transformed into a format directly compatible with the TV software.

This rationale for this two step process is that by initially selecting from the ITD data those items of value for the TV views to be generated later, the second step can concentrate solely on the data specific to one view type only. This second step will process data much more quickly than if it had to deal with the entire ITD data set.

Both intermediate and final file formats are defined below:

A. Vegetation Polygon Intermediate File

The format for the vegetation polygon attribute data file [$FEAT_OVERLAY:SSS_POLY_PAT] is defined below:

Field	Type	Content
POLY_REC_NO	Integer	Sequentially assigned beginning at 1
VGC	Integer	ITD VGC Attribute
BUD	Integer	ITD BUD Attribute
CCR	Integer	ITD CCR Attribute
SD1	Integer	ITD SD1 Attribute
SD2	Integer	ITD SD2 Attribute
TS1	Integer	ITD TS1 Attribute
VH1	Integer	ITD VH1 Attribute
VR1	Integer	ITD VR1 Attribute
MIN_LAT	Integer	Minimum latitude of feature in 1/100 arc-sec
MIN_LON	Integer	Minimum longitude of feature in 1/100 arc-sec
MAX_LAT	Integer	Maximum latitude of feature in 1/100 arc-sec
MAX_LON	Integer	Maximum longitude of feature in 1/100 arc-sec
COORD_REC	Integer	Number of records in the SEG file (see following section)

In addition to the attribute file, defined above, the vegetation polygon outlines are described by the file $FEAT_OVERLAY:SSS_POLY_COORDS. This file consists of Ada records, with the number of records corresponding to those required to describe the total set of polygons. The format of these records is (each field is a 4-byte integer):

For Record 1:
    Field #1: POLY_REC_NO (as described above)
    Field #2: Number of arcs defining the outline of the feature

For Record 2:
    Field #1: Sentinel (999999999)
    Field #2: Number of points in the arc [repeated for each arc)

For Record 3:
    Field #1: Latitude (1/100 arc-sec) of first point of arc
    Field #2: Longitude (1/100 arc-sec) of first point of arc

For Record 4:
    Field #1: Same as Field #1 of Record 3 for second point
    Field #2: Same as Field #2 of Record 3 for second point

etc.

B. Surface Drainage Line Intermediate File

The format for the surface drainage line attribute data file [$FEAT_OVERLAY:SDR_LINE_AAT] is defined below:

Field	Type	Content
LINE_REC_NO	Integer	Sequentially assigned beginning at 1
OFC5	Integer	ITD OFC Attribute
BMC	Integer	ITD BMC Attribute
DW1	Integer	ITD DW1 Attribute
EXS	Integer	ITD EXS Attribute
HFC	Integer	ITD HFC Attribute
HYC	Integer	ITD HYC Attribute
LEN	Integer	ITD LEN Attribute
TID	Integer	ITD TID Attribute
WID	Integer	ITD WID Attribute
WVA	Integer	ITD WVA Attribute
MIN_LAT	Integer	Minimum latitude of feature in 1/100 arc-sec
MIN_LON	Integer	Minimum longitude of feature in 1/100 arc-sec
MAX_LAT	Integer	Maximum latitude of feature in 1/100 arc-sec
MAX_LON	Integer	Maximum longitude of feature in 1/100 arc-sec
COORD_RECS	Integer	Number of records in the SEG file (see following section)

In addition to the attribute file, defined above, the surface drainage lines are described by the file $FEAT_OVERLAY:SDR_LINE_COORDS. This file consists of a series of two word Ada records, with the number of records corresponding to those required to describe the total set of polygons. The format of these records is (each field is a 4-byte integer):

For Record 1:
    Field #1: LINE_REC_NO (as described above)
    Field #2: Number of arcs defining the outline of the feature

For Record 2:
Field #1: Sentinel (999999999)
Field #2: Number of points in the arc [repeated for each arc)

For Record 3:
Field #1: Latitude (1/100 arc-sec) of first point of arc
Field #2: Longitude (1/100 arc-sec) of first point of arc

For Record 4:
Field #1: Same as Field #1 of Record 3 for second point
Field #2: Same as Field #2 of Record 3 for second point

etc.

C. Surface Drainage Polygon Intermediate File

The format for the surface drainage polygon attribute data file [$FEAT_OVERLAY:SDR_POLY_PAT] is defined below:

Field	Type	Content
POLY_REC_NO	Integer	Sequentially assigned beginning at 1
BMW	Integer	ITD BMW Attribute
DW1	Integer	ITD DW1 Attribute
GW4	Integer	ITD GW4 Attribute
HFC	Integer	ITD HFC Attribute
HYC	Integer	ITD HYC Attribute
LEN	Integer	ITD LEN Attribute
TID	Integer	ITD TID Attribute
WID	Integer	ITD WID Attribute
WVA	Integer	ITD WVA Attribute
MIN_LAT	Integer	Minimum latitude of feature in 1/100 arc-sec
MIN_LON	Integer	Minimum longitude of feature in 1/100 arc-sec
MAX_LAT	Integer	Maximum latitude of feature in 1/100 arc-sec
MAX_LON	Integer	Maximum longitude of feature in 1/100 arc-sec
COORD_RECS	Integer	Number of records in the SEG file (see following section)

In addition to the attribute file, defined above, the surface drainage polygons are described by the file $FEAT_OVERLAY:SDR_POLY_COORDS. This file consists of a series of two word Ada records, with the number of records corresponding to those required to describe the total set of polygons. The format of these records is (each field is a 4-byte integer):

For Record 1:
    Field #1: POLY_REC_NO (as described above)
    Field #2: Number of arcs defining the outline of the feature

For Record 2:
    Field #1: Sentinel (999999999)
    Field #2: Number of points in the arc [repeated for each arc)

For Record 3:
    Field #1: Latitude (1/100 arc-sec) of first point of arc
    Field #2: Longitude (1/100 arc-sec) of first point of arc

For Record 4:
    Field #1: Same as Field #1 of Record 3 for second point
    Field #2: Same as Field #2 of Record 3 for second point

etc.

D. Transportation Line Data Intermediate File

The format for the transportation line attribute data file [$FEAT_OVERLAY:TRS_LINE_AAT] is defined below:

Field	Type	Content
LINE_REC_NO	Integer	Sequentially assigned beginning at 1
OFC4	Integer	ITD OFC4 Attribute
ACC	Integer	ITD ACC Attribute
EXS	Integer	ITD EXS Attribute
MED	Integer	ITD MED Attribute
WTC	Integer	ITD WTC Attribute
LTN	Integer	ITD LTN Attribute
RSA	Integer	ITD RSA Attribute
RPS	Integer	ITD RPS Attribute
LEN	Integer	ITD LEN Attribute
NOS	Integer	ITD NOS Attribute
MCP	Integer	ITD MCP Attribute
WTW	Float	ITD WTW Attribute
MIN_LAT	Integer	Minimum latitude of feature in 1/100 arc-sec
MIN_LON	Integer	Minimum longitude of feature in 1/100 arc-sec
MAX_LAT	Integer	Maximum latitude of feature in 1/100 arc-sec
MAX_LON	Integer	Maximum longitude of feature in 1/100 arc-sec
COORD_RECS	Integer	Number of records in the SEG file (see following section)

In addition to the attribute file, defined above, the transportation line features are described by the file $FEAT_OVERLAY:TRS_LINE_COORDS. This file consists of a series of two word Ada records, with the number of records corresponding to those required to describe the total set of lines. The format of these records is (each field is a 4-byte integer):

For Record 1:
    Field #1: LINE_REC_NO (as described above)
    Field #2: Number of arcs defining the outline of the feature

For Record 2:
    Field #1: Sentinel (999999999)
    Field #2: Number of points in the arc [repeated for each arc)

For Record 3:
    Field #1: Latitude (1/100 arc-sec) of first point of arc
    Field #2: Longitude (1/100 arc-sec) of first point of arc

For Record 4:
    Field #1: Same as Field #1 of Record 3 for second point
    Field #2: Same as Field #2 of Record 3 for second point

etc.

E. DTSS Masked Area Plot Data Intermediate File

The format for the Masked Area Plot attribute data file [$FEAT_OVERLAY:MAP_LINE_AAT] is defined below:

Field	Type	Content
LINE_REC_NO	Integer	Sequentially assigned beginning at 1
CODE	Integer	Indicates type of data [range ring, range line, border]
MIN_LAT	Integer	Minimum Latitude of feature in 1/100 arc-secs
MIN_LONG	Integer	Minimum Longitude of feature in 1/100 arc-secs
MAX_LAT	Integer	Maximum Latitude of feature in 1/100 arc-secs
MAX_LONG	Integer	Maximum Longitude of feature in 1/100 arc-secs
COORD_RECS	Integer	Number of records in the SEG file (see following section)

In addition to the attribute file, defined above, the Masked Area Plot lines are described by the file $FEAT_OVERLAY:MAP_LINE_COORDS. This file consists of a series of two word Ada records, with the number of records corresponding to those required to describe the total set of lines. The format of these records is (each field is a 4-byte integer):

For Record 1:
    Field #1: LINE_REC_NO (as described above)
    Field #2: Number of arcs defining the outline of the product

For Record 2:
    Field #1: Sentinel (999999999)
    Field #2: Number of points in the arc [repeated for each arc)

For Record 3:
    Field #1: Latitude (1/100 arc-sec) of first point of arc
    Field #2: Longitude (1/100 arc-sec) of first point of arc

For Record 4:
Field #1: Same as Field #1 of Record 3 for second point
Field #2: Same as Field #2 of Record 3 for second point

etc.

F. DTSS Target Acquisition Model Data Intermediate File

The format for the Target Acquisition Model Data attribute data file [$FEAT_OVERLAY:TAM_LINE_AAT] is defined below:

Field	Type	Content
LINE_REC_NO	Integer	Sequentially assigned beginning at 1
CODE	Integer	Indicates type of data [range line, range ring, border]
MIN_LAT	Integer	Minimum Latitude of feature in 1/100 arc-secs
MIN_LONG	Integer	Minimum Longitude of feature in 1/100 arc-secs
MAX_LAT	Integer	Maximum Latitude of feature in 1/100 arc-secs
MAX_LONG	Integer	Maximum Longitude of feature in 1/100 arc-secs
COORD_RECS	Integer	Number of records in the SEG file (see following section)

In addition to the attribute file, defined above, the Target Acquisition Model plot lines are described by the file $FEAT_OVERLAY:TAM_LINE_COORDS. This file consists of a series of two word Ada records, with the number of records corresponding to those required to describe the total set of lines. The format of these records is (each field is a 4-byte integer):

For Record 1:
Field #1: LINE_REC_NO (as described above)
Field #2: Number of arcs defining the outline of the product

For Record 2:
Field #1: Sentinel (999999999)
Field #2: Number of points in the arc [repeated for each arc)

For Record 3:
Field #1: Latitude (1/100 arc-sec) of first point of arc
Field #2: Longitude (1/100 arc-sec) of first point of arc

For Record 4:
Field #1: Same as Field #1 of Record 3 for second point
Field #2: Same as Field #2 of Record 3 for second point

etc.

G. Intermediate Information File

As each intermediate file is generated, the number of features processed is used to update the Intermediate Information FIle [$FEAT_OVERLAY_:INT_INF_FILE. This file identifies the total number of features present for each feature type. The format for the file is defined below:

Field Identifier	Type	Content
NO_OF_SSV_POLY_RECS	Integer	No of records in SSV poly file
NO_OF_TRS_POLY_RECS	Integer	No of records in TRS poly file
NO_OF_TRS_LINE_RECS	Integer	No of records in TRS line file
NO_OF_TRS_POINT_RECS	Integer	No of records in TRS point file
NO_OF_SDR_POLY_RECS	Integer	No of records in SDR poly file
NO_OF_SDR_LINE_RECS	Integer	No of records in SDR line file
NO_OF_OBS_POLY_RECS	Integer	No of records in OBS poly file
NO_OF_OBS_LINE_RECS	Integer	No of records in OBS line file
NO_OF_OBS_POINT_RECS	Integer	No of records in OBS point file
NO_OF_BCL_POLY_RECS	Integer	No of records in BCL poly file
NO_OF_BCL_LINE_RECS	Integer	No of records in BCL poly file
NO_OF_BCL_POINT_RECS	Integer	No of records in BCL point file
NO_OF_MAP_LINE_RECS	Integer	No of records in MAP line file
NO_OF_TAM_LINE_RECS	Integer	No of records in TAM line file

Note that some of these feature types are not currently processed into intermediate file format by the terrain visualization software. They are included in the table for future utilization. Entries in the file for these features are currently null values.

H. Final Feature Overlay Files

Once the features have been processed into the intermediate file format, they are available for final transformation so that they can be used in inserting model data into the perspective view. The record types used in this final feature overlay file are described below:

1. FEATURE_CLASS_AND_EXTENT record

This record defines the feature type as either vector line data, tree line data, or an end of the data file sentinel. The format is:

Field Identifier	Type	Content
FEAT_CLASS_AND_EXTENT	Ada Enumeration	This field can contain: VECT_LINE TREE_LINE NO_MORE
MIN_LAT	Integer	Feature extent south latitude in 1/100 arc-sec increments
MIN_LON	Integer	Feature extent west longitude in 1/100 arc-sec increments
MAX_LAT	Integer	Feature extent north latitude in 1/100 arc-sec increments
MAX_LON	Integer	Feature extent east longitude in 1/100 arc-sec increments
NO_OF_SEG_RECS	Integer	The number of line segments

For the two types of model data unique line record/segment record pairs are defined. The pair definitions for vector line versus tree line follow:

2. Vector Line Header Record [VECT_LINE_HEAD_REC]

Field Identifier	Type	Content
NO_OF_VECT_SEGMENTS	Integer	Number of segments in the vector track
VECT_TYPE	Integer	Line segment color [1:Black, 2:Magenta, 3:Red, 4:Yellow, 5:Green, 6:Cyan, 7:Blue, 20:White]
VECT_WIDTH	Integer	Width of the vector in meters
START_LAT	Float	Vector starting latitude in arc-sec
START_LON	Float	Vector starting longitude in arc-sec

3. Vector Segment Record [VL_SEG_REC]

Field Identifier	Type	Content
SEG_DIST	Integer	Length of this segment in meters
SEG_DIR	Integer	Heading of this segment in degrees [-180 to +180]
INTEGER_DIST	Integer	Maximum interval distance in meters

4. Tree Line Record [TREE_LINE_REC]

Field Identifier	Type	Content
NO_OF_TL_SEGMENTS	Integer	Number of segments in the tree line
START_LAT	Float	Vector starting latitude in arc-sec
START_LON	Float	Vector starting longitude in arc-sec
PERCENT_BAREBRANCHES	Integer	Average percentage bare trees randomly selected
PERCENT_GREENTREES	Integer	Average percentage green trees randomly selected
PERCENT_FALLTREES	Integer	Average percentage fall trees randomly selected
PERCENT_LOWPINES	Integer	Average percentage low pines trees randomly selected
PERCENT_TALLPINES	Integer	Average percentage bare trees randomly selected
AVERAGE_HEIGHT	Integer	Average height of the trees (meters)
HEIGHT_VARIATION	Integer	Maximum height variation from average (meters)
AVERAGE_SPACING	Integer	Average spacing between trees (meters)
SPACING_VARIATION	Integer	Maximum spacing variation (meters)

5. Tree Line Segment Record [TL_SEG_REC]

Field Identifier	Type	Content
SEG_DIST	Integer	Length of this segment in meters
SEG_DIR	Integer	Heading of this segment in degrees [-180 to +180]

4.2.2 Processing

4.2.2.1 Affine Transform

This section provides a discussion of the affine transform used for the geographic rectification of the imagery and digitized-map data sources.

In order to overlay any imagery type data, it is first necessary to convert the image data into a map coordinate frame. The algorithm used to accomplish this task was the affine transform. This transform can model six types of distortion in an image: 1.) translation in x, 2.) translation in y, 3.) scale changes in x, 4.) scale changes in y, 5.) skew, and 6.) rotation. It is not necessary to directly know these distortions in order to develop the transformation coefficients, since they are developed by an iterative mathematical process using measured control points. The affine transformation does not remove image layover distortion associated with vertical height variations. A discussion of the transform and the method of implementing it follows.

The form of the transform is:

$$x_i = A_0 + A_1 x_m + A_2 y_m$$

$$y_i = B_0 + B_1 x_m + B_2 y_m$$

x_i and y_i	are in the image coordinate frame
x_m and y_m	are in the map coordinate frame
$A_0, A_1, A_2, B_0, B_1, B_2$	are the transformation coefficients

In order to develop the coefficients, a number of control points are selected for each image. Once the control points were selected, the next step in the process is to measure the geographic position for each control point (from a map, for example) and then to identify where each of these control points is in the imagery. In the case of a map, the tic marks are used (the geographic positions of these is of course already known) as a primary reference providing the necessary information for the determination of the precise geographic location of the control information.

The affine-transform algorithm utilizes an iterative process to develop the coefficients. In each pass, the coefficients are computed, using a least-squares multiple-regression technique; then the transform is applied to each of the control points. The average-root-mean square (RMS) error between the measured pixel position and the predicted pixel position is computed for the remaining control points. If the average error is below a preselected threshold, the process is finished; otherwise, the point with the worst error is deleted from the remaining points, and the next pass begins. A more detailed discussion of the coefficient computation follows.

The following notation will be used:

Y	x or y location of the control point in the image
X_1	easting coordinate of the control point
X_2	northing coordinate of the control point

4.2.2.1.1 Coefficient Derivation

A number of steps are necessary in order to derive the affine transform coefficients. The following steps are performed for both the x and y positions for the control points in order to determine both the A and B coefficients. Only the derivation of the A coefficients is shown since the determination of B coefficients utilizes the same procedure.

4.2.2.1.1.1 Adjust values

The northing and easting coordinate points are adjusted to denote deltas from the minimum values of the batch of control points; this is done to prevent the values from becoming too large and losing precision. This operation is represented by:

$$X_{n\ adjusted} = X_{n\ original} - \text{MIN}\ X$$

4.2.2.1.1.2 Derive X^TX Matrix

The coefficients for the matrix X^TX are derived from the sum of squares computation:

Given:

$$X^TX = \begin{pmatrix} a_{xx} & b_{xx} \\ c_{xx} & d_{xx} \end{pmatrix}$$

Then the coefficients are derived as follows:

$$a_{xx} = \sum_{i=1}^{n} X_{1i}^2 - \frac{1}{n}\left(\sum_{i=1}^{n} X_{1i}\right)^2$$

$$d_{xx} = \sum_{i=1}^{n} X_{2i}^2 - \frac{1}{n}\left(\sum_{i=1}^{n} X_{2i}\right)^2$$

$$b_{xx} = c_{xx} = \sum_{i=1}^{n} X_{1i}X_{2i} - \frac{1}{n}\left(\sum_{i=1}^{n} X_{1i}\right)\left(\sum_{i=1}^{n} X_{2i}\right)$$

4.2.2.1.1.3 Derive X^TY Matrix

The coefficients of the matrix X^TY are computed from the covariance between Y and X_1 and the covariance between Y and X_2.

Given:

$$X^TY = \begin{pmatrix} a_{xy} \\ b_{xy} \end{pmatrix}$$

Then the coefficients are derived as follows:

$$a_{xy} = \sum_{i=1}^{n} X_{1i}Y_i - \frac{1}{n}\left(\sum_{i=1}^{n} X_{1i}\right)\left(\sum_{i=1}^{n} Y_i\right)$$

$$b_{xy} = \sum_{i=1}^{n} X_{2i} Y_i - \frac{1}{n}\left(\sum_{i=1}^{n} X_{2i}\right)\left(\sum_{i=1}^{n} Y_i\right)$$

4.2.2.1.1.4 Derive Inverse of $X^T X$ Matrix

The inverse of the matrix $X^T Y$ is then computed

Given:

$$X^T X = \begin{pmatrix} a_{xx} & b_{xx} \\ c_{xx} & d_{xx} \end{pmatrix}$$

Then the determinant is:

$$det A = |X^T X| = |ad - cb|$$

and the inverse is:

$$(X^T Y)^{-1} = \frac{1}{det A}\begin{pmatrix} d_{xx} & -b_{xx} \\ -c_{xx} & a_{xx} \end{pmatrix}$$

and is renotated as follows for convenience:

$$(X^T Y)^{-1} = \begin{pmatrix} a_{xxi} & b_{xxi} \\ c_{xxi} & d_{xxi} \end{pmatrix}$$

4.2.2.1.1.5 Compute Coefficients

Finally, the coefficients are derived:

$$\begin{pmatrix} A_1 \\ A_2 \end{pmatrix} = (X^T X)^{-1}(X^T Y)$$

$$\begin{pmatrix} A_1 \\ A_2 \end{pmatrix} = \begin{pmatrix} a_{xxi} & b_{xxi} \\ c_{xxi} & d_{xxi} \end{pmatrix}\begin{pmatrix} a_{xy} \\ b_{xy} \end{pmatrix}$$

$$A_0 = \overline{Y} - \left(\sum_{i=1}^{2} a_i \overline{X}_i\right)$$

4.2.2.1.2 RMS Error

The following equation is used in order to determine the RMS error for each of the control points.

$$Error_{RMS} = \sqrt{(x_{computed} - x_{original})^2 + (y_{computed} - y_{original})^2}$$

4.3 General Parametric Input

The area most important to human satisfaction with a particular software product is the conversational interface - the Man Machine Interface (MMI). A powerful software capability may be relegated to disuse by inattention to

human factors. Conversely, a software system not quite as rich in capability or underlying hardware support may greatly offset these negatives through careful, attentive, MMI design.

While the design of an effective user interface remains partially an artistic endeavor, scientific characterizations of what is "good" are progressing. Ergonomically based principles for effective terrain-visualization MMI design include the following:

* Proper Prompting - Clear descriptions of MMI expectations; a hierarchical help facility allowing for a wide spectrum of terrain analyst proficiency levels.
* Interactivity - Positive and negative feedbacks; a planned and programmed dialogue specifically for the terrain analyst audience.
* Error Toleration - Provision for a variety of levels of accommodation of erroneous input; correction of the individual character, parameter, or logical parameter grouping.
* Response Time - Timing requirements in accordance with MIL-STD-1472C, Notice 2.
* Consistency - Obedience to fundamental conceptual model, syntax, semantics, and display formats.
* Structured Display - Logical groupings of parameters; visual encodings which amplify the terrain analysts capability to discern related items.

The MMI developed for the DTSS provided an interactive conversation definition tool which incorporates the principles outlined above. The General Parametric Input function utilizes the DTSS MMI to allow specification of parameters which control the execution of the progressive refinement pipeline. Input parameters are selectable by the analyst in a sequence of menus generated on the GPX display. Selectable parameters include:

* Progressive refinement initial phase - What is the starting step in the progressive refinement chain?
* Progressive refinement terminating phase - What is the ending step in the progressive refinement chain?
* Continuous-Surface Display display type - terrain shading only,
* Continuous-Surface Display draped data source - Digital map data, terrain analysis products and source data, or remotely sensed imagery.
* Critical element identification - the location and description of elements within the perspective area of tactical or strategic interest.
* Terrain fractalization - Surface noise, tension, and interpolation interval.

4.4 Interactive Parametric Input

The visualization of a 3D location and orientation is generally difficult for terrain analysts. Selection of the perspective 3D orientation parameters offer a particularly challenging area where graphical displays and icons may be used with major benefit. Data presented pictorially harnesses the well developed eye-brain recognition mechanism and allows us to perceive and process data more rapidly and efficiently.

Analyst modification of perspective parameters should be natural. A locator type device should be used for pie icon positioning. For PAWS implementation, the mouse was used for graphical manipulation.

With a graphical interaction technique in place, an analyst is better equipped to explore and visualize the terrain environment of the workspace, and the progressive-refinement rendering hierarchy is better controlled and directed.

4.4.1 Description

The Interactive Parametric Input task is a graphical interface which allows the operator to manipulate geometrical icons to select three-dimensional parametric values. To reiterate, typical perspective-view implementations allow for the following parametric inputs:

Viewing position (geographic location and elevation)

View Direction

Field of View

Declination angle

Observation range

Vertical exaggeration factor of terrain elevation values

Sun Position and Declination angle

Terrain analysis is accomplished within the context of a workspace. A workspace is a geographically rectangular area, generally associated with mapsheet boundaries, that establishes the limits of the terrain analysis data of interest. A rectangle, drawn with the workspace aspect ratio, is used as a representation of the workspace areal extent. This workspace-rectangle may be filled with location referencing information such as a terrain sample grid, terrain analysis source data, or continent-country outline display.

Figure 4. illustrates a composite image of what the analyst might see. A separate window indicates the current parametric settings in text form and a sun icon is positioned along the border of the workspace rectangle to indicate relative sun position for shading. The pie or rectangle icon is non-destructively drawn within the workspace rectangle. Each of these areas will be further discussed in the following sections.

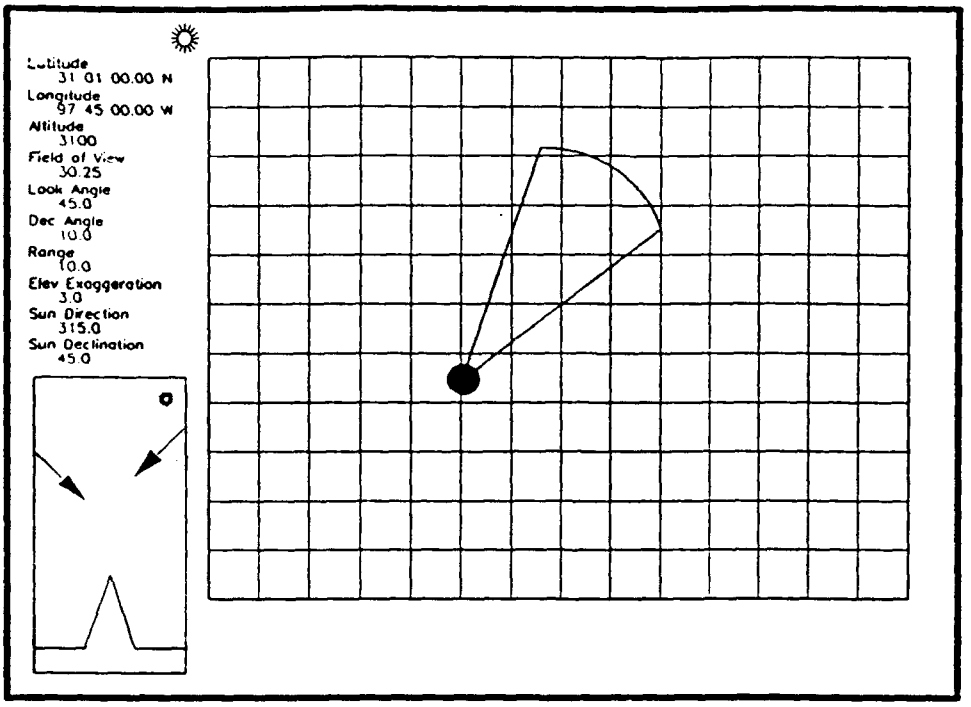


Figure 4. Operator Interaction Screen

4.4.2 Inputs and Outputs

In order to begin the Interactive Parametric Input, a number of operator inputs are necessary. These inputs will be defined in the MMI, and will include 1.) Viewer Position (latitude, longitude, altitude), 2.) Viewer Look Parameters (look direction, field of view, declination angle, observation range), 3.) Background Display Type (for location referencing). The supported background types are: 1.) Terrain Sample Grid, 2.) Terrain Analysis Source Data (Drainage and Roads), 3.) Scanned map products and 4.) Imagery.

The mouse and the keyboard will be used for the operator input devices. The operator will select a point to move (for example, the viewer location), and will then drag the icon to the desired location.

The interactive mesh program will be notified when parameters are updated.

4.4.3 Graphical Representation of Operator Input

4.4.3.1 Viewer Position Icon

The viewer position icon will be overlaid on the background data at the appropriate location. This icon (see Figure 5.) portrays the following information: 1.) Viewer Position (geographic location), 2.) Viewer Look Direction (compass heading), 3.) Viewer Field of View (angular extent of viewing aperture), and 4.) Observation Range (prediction area radius).

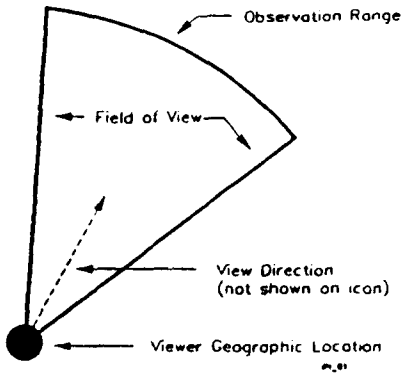


Figure 5. Viewer Position Icon

The apex is positioned at the geographic location of the viewing position. The sides of the pie indicate the field of view and view direction. The bounding arc is located at a distance which reflects the observation range.

4.4.3.2 Elevation Icon

A separate icon depicts parameters which are related to the elevation. This elevation icon will be displayed on the lower left corner of the viewing window. This icon (see Figure 6.) portrays the following information: 1.) Vertical Exaggeration Factor of terrain elevation values, 2.) Declination angle (observer vertical orientation), 3.) Viewing Position (altitude with respect to ground level (AGL)), and 4.) Sun Angle.

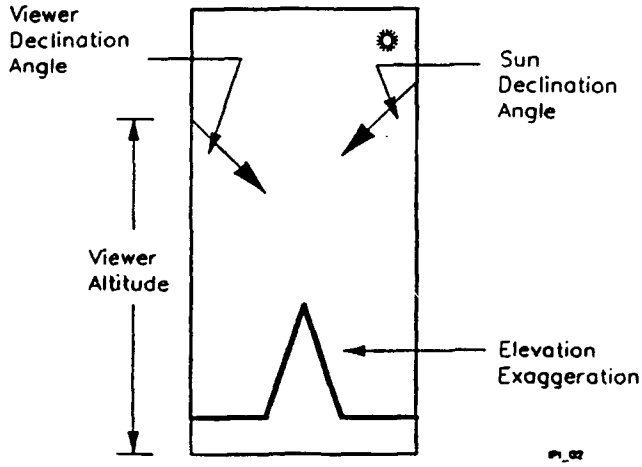


Figure 6. Elevation Icon

An arrow suspended from a point to the left and above a horizontal line indicates viewing position elevation and declination angle. An arrow suspended to the right and above the horizontal line indicates the declination angle of the sun. The height of a pyramid on the horizontal line indicates the current vertical exaggeration factor.

4.4.3.3 Sun Direction Icon

This icon will be positioned somewhere along the outside edge of the location referencing grid, and will denote the azimuth angle of the light source with respect to the data base. This is illustrated in Figure 4.

4.4.4 Text Representation of Operator Input

The text representation of the operator input values will be displayed in the upper left corner of the viewing window. This is illustrated in Figure 4.

4.4.5 Processing Steps

As stated above, the initial parameters for this step are defined in the MMI, so the first step is to retrieve these parameters from the menu data files.

The next step is to display the background information for the operator to use as positional referencing.

The parameter values in text form are written to the graphical screen.

The location parameters are then transformed to screen coordinates. The screen coordinates are determined by geographic extents of the displayed background area and its position on the graphical screen. These are then drawn in a graphical form as described above.

Input is then received from the operator. When a parameter is changed, the screen display is updated. Concurrently, the interactive mesh display is receiving the newly defined parameters in order to generate a new mesh display. When the operator is finished modifying the input parameters, the middle mousebutton is depressed. At this point, the input parameters are finalized and the progressive refinement chain begins.

4.5 Interactive Mesh Generation

The responsibility of achieving near-real-time display update rests on the image generation algorithms at this level. The output of this level is a wire-mesh representation of terrain that is within the viewing volume without elimination of hidden line segments. A sequence of steps are performed:

1. Determine the potentially visible portion of the terrain grid. If the entire area can not be displayed in a minimally acceptable time, an appropriate subset of the potentially visible area is initially displayed.
2. Access and transform the terrain elevation data from the geographic reference frame to device coordinates.
3. Pair neighboring transformed coordinates to form line segments and pass visible portions of line segments to the graphics processor for display.

The preceding discussion assumes a very rudimentary graphics processor, capable only of line-segment drawing. If graphics processor capabilities include change of basis transformations and/or line clipping, items 2 and 3 above should utilize the processor to enhance the speed of display.

The amount of data to initially display is dependent on the viewing parameters and may be too great in volume to insure a sufficient level of output performance. As mentioned in item 1 above, the algorithm should be structured so that a representative portion of the terrain mesh is displayed at interactive rates. The particular portion to be displayed might be achieved by skipping data (e.g. every other point, every third point, etc.). If the initial display must be approximated, the algorithm proceeds to complete the display after the initial approximation is made. If at any time during the process, the orientation parameters are updated by the operator, the mesh display process must be restarted.

4.5.1 Description

The Interactive Mesh Display is terrain represented as a wire-frame-mesh perspective view, representing the connection of the tops of elevation grid posts.

4.5.2 Inputs and Outputs

A number of input parameters are necessary in order to execute this step in the progressive-refinement chain. Among these are: 1.) Viewer positional information (latitude, longitude, altitude), 2.) Viewer look information (look direction, field of view, declination angle, observation range). These parameters are determined during the Interactive Parametric Input stage.

Aside from the control parameters, elevation data is the only input data necessary; this is currently DTED data.

The output of this step in the progressive-refinement chain is the mesh display, with no hidden lines removed, as illustrated in Figure 7.

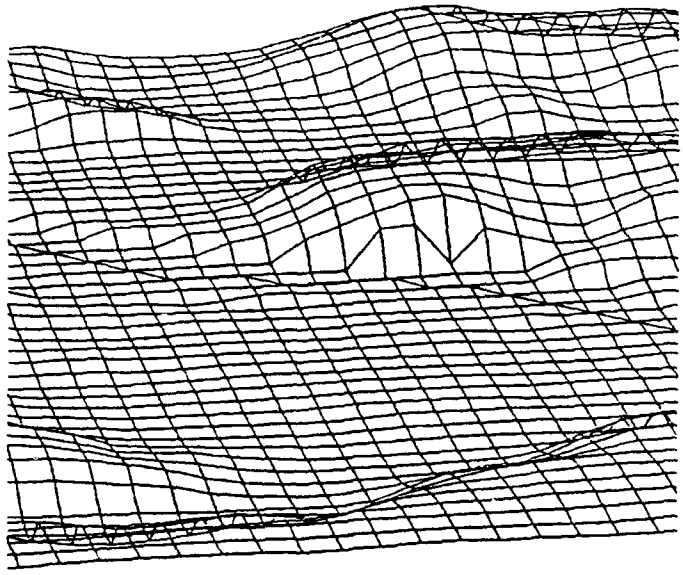


Figure 7. Mesh Display, Without Hidden-Line Removal

4.5.3 Processing Steps

One of the first steps in order to maintain very interactive displays is to sample the display to achieve the interactive rates desired. First the number of points included in the display is determined by calculating the extents of the pie-shaped coverage of the display area, and computing the number of elevation posts. A maximum-point-count threshold is predetermined by the following method: (1) A standard maximum display time (arbitrarily chosen to be one second) was defined and the rate per perspective point was determined. (2) If the number of points requested exceeded the threshold (thereby causing the allotted time to be exceeded), sampling must occur. (3) The ratio of the number of points requested in the display to the maximum number of points determines the sampling interval.

Note that this method assumes that the interactive display rate is dominated by perspective transform time. If this is not the case [e.g., large areas of elevation data must be accessed before the data is sampled, thereby causing data access to dominate the time], longer display times will necessarily result.

The input parameters are used to define the transformation coefficients necessary to convert the geographic coordinate of each point to screen coordinates. The following explains the projection of the view window onto the database reference plane.

The following transformations are made: (1) The window is projected onto an aligned reference plane (XR, YR) where YR is in the along-range direction and where the Line of Sight (LOS) intersects the reference plane at its origin. (2) The results obtained can easily then be transformed to an actual database coordinate system through ordinary coordinate translation and rotation operations. The variables are defined as follows:

f_0 = observer to view window perpendicular distance

SL_0 = slant range from observer (viewpoint) to the origin of the reference plane

Θo = depression angle of the LOS (line along which SL_0 is measured) relative the horizontal

WS = view window width

HS = view window height

αv = view window vertical viewing angle

where f_0, SL_0, WH, and HS are all in same length units .

where Θ_0 and α_V are in angle units

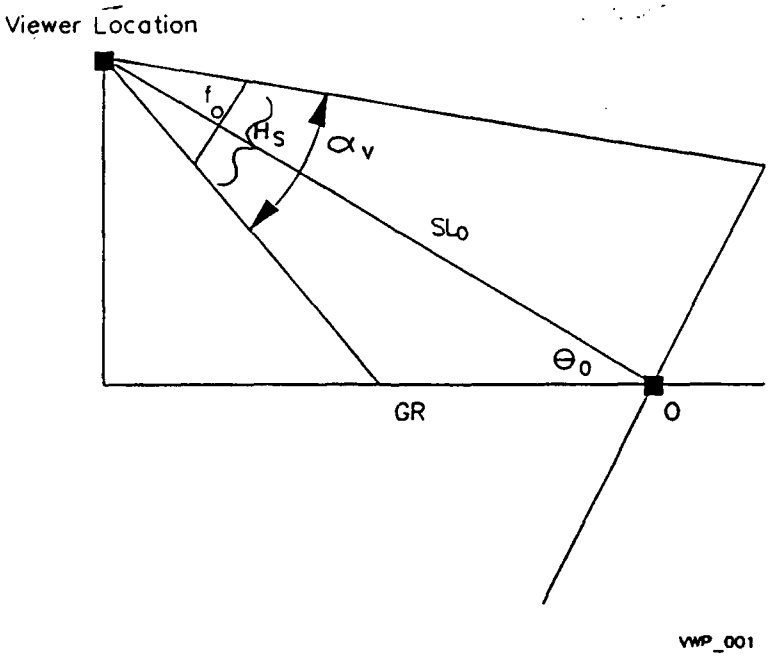


Figure 8. View Window Projection

Calculate the intermediate variables

$$HC = \left(\frac{SL_o}{f_o}\right) * HS$$

$$A = \left(\frac{HC}{2}\right) * \cot\Theta_o$$

$$B = A * \left[\frac{\sin\alpha_v}{\sin(\Theta_o - \alpha_v)}\right]$$

$$C = A * \left[\frac{\sin\alpha_v}{\sin(180° - (\Theta_o + \alpha_v))}\right]$$

$$E = \left(\frac{HC}{2}\right) * \left(\frac{1}{\sin\Theta_o}\right)$$

The intermediate variables are introduced in order to package constants involving complex trigonometric functions, not needing to be recomputed for each point, into a form which will minimize computation.

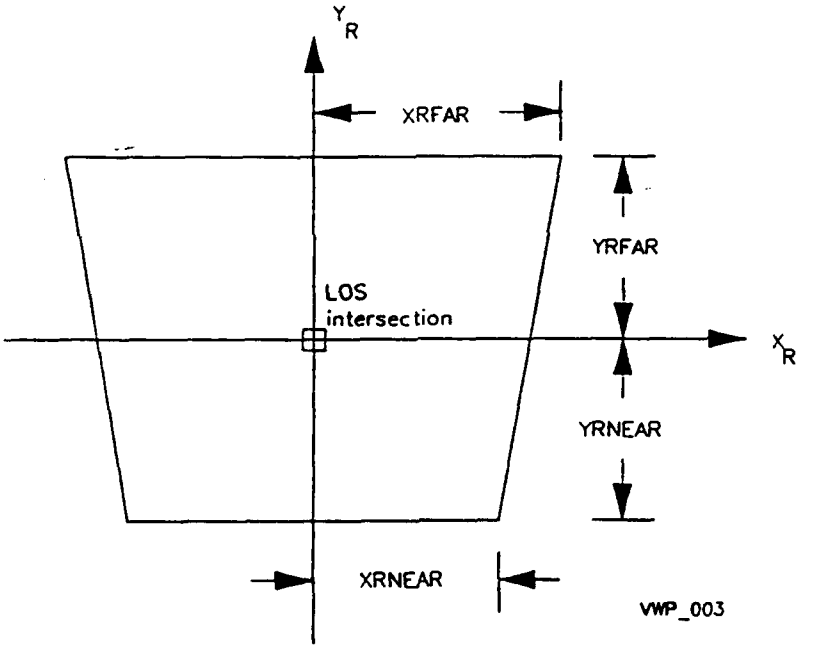


Figure 9. View Window Projection Extents

Calculate the projection dimensions:

$$YRFAR = E + B$$

$$YRNEAR = E - C$$

$$XRFAR = \left(\frac{WS}{2f_o}\right)(SL_o + B\cos\Theta_o + A)$$

$$XRNEAR = \left(\frac{WS}{2f_o}\right)(SL_o + C\cos\Theta_o - A)$$

Graphics performance requirements demand that the graphics output bypass NovaGKS and interface directly with the Parallax videographic processor. By double-buffering graphics commands, the VAX and Parallax may operate in parallel (concurrently) and maximize system performance. This algorithm characteristically generates many short vectors. Unfortunately the Parallax videographics processor has no "relative" line drawing capability. The processor does have a multiple line drawing command which identifies a point count and packs the x, y screen coordinates into one point per longword (32 bits). This leads to a significant reduction in data transfer between VAX and Parallax and faster Parallax processing. Because the drawing order for this display is not important, the display is always constructed along lines of constant x, and then lines of constant y, thus allowing many vectors to be sent to display at one time. Figure 10 illustrates the vectors and the ordering to allow multiple vector draws to be accomplished with a single call to the display computer. Careful look-up table modification, see Figure 11, allows for sequential frames to be constructed within the Parallax frame buffer as the operator is viewing the previous image. Upon image completion, screen update may occur almost instantaneously by look-up table changes.

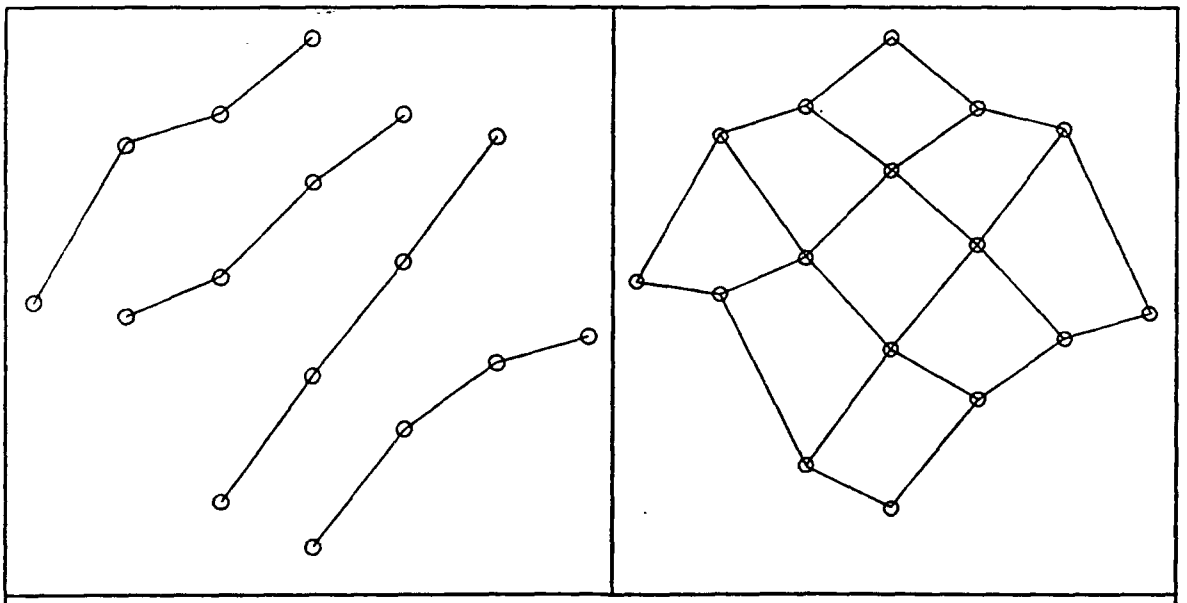


Figure 10. Line Drawing Order for Interactive Mesh

Plane 1	Plane 2	LUT	Screen	Operation
		Plane 1 Visible		Draw first image
		Plane 1 Visible		Draw second image on plane 2
		Plane 2 Visible		Flip look-up table
		Plane 2 Visible		Erase first image on plane 1
		Plane 2 Visible		Draw next image on plane 1
		Plane 1 Visible		Flip look-up table

Figure 11. Image Plane and Look-Up Table Management

4.6 Hidden Line Removal

The first step toward realism involves removal of those line segments which are obscured from view by other parts of the terrain. The improvement in terrain perception which this step allows can be understood by comparing Figure 7 (without hidden line removal) to Figure 12 (with hidden line removal). The floating-horizon hidden line removal algorithm serves well in this role.

4.6.1 Description

The algorithm examines points in order of increasing distance from the viewing position (front to back). Line segments combine to form a horizon definition which subsequent line segments must rise above to be visible. The horizon definition is updated to include line segments which are visible - causing the horizon to "float" upward as the algorithm proceeds.

One of the fundamental goals of progressive refinement is to perform image refinement such that the transition to realism is as inconspicuous as possible. Transition to a hidden line level of realism occurs most naturally when algorithmic involvement appears to be limited to erasure of the line segments which are hidden. The resulting display should not contain gaps in the visible line segments where a hidden line was erased.

To achieve this, separate bit-planes and writing modes may be used. Assume that the interactive mesh display is in bit-plane 1 and the hidden mesh display will be written to bit-plane 2. As visible lines are written to bit-plane 2, they are written with a logical "OR" so that data within plane 1 is not destroyed. Invisible lines are written into bit-plane 1 and effectively turn-off the pixels that are hidden with a logical "AND" operation. The desired transition occurs naturally and inconspicuously.

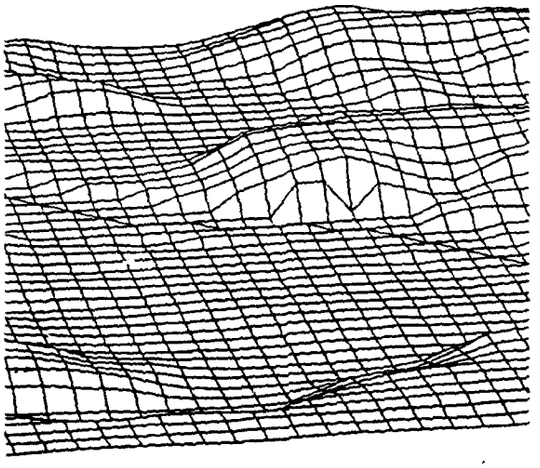


Figure 12. Mesh Display, with Hidden Line Removal

4.6.2 Inputs and Outputs

The only inputs necessary for this display are 1.) the viewer look direction, and 2.) the transformed screen coordinates. The look direction is necessary in order to determine the proper drawing order for the use of the floating-horizon algorithm. The screen coordinates are, of course, used to draw the image. Note that the transformed coordinates have already been computed during the mesh display.

4.6.3 Processing Steps

The first processing step is to access the screen coordinates. The next step is to determine which drawing order is to be used. This technique is discussed separately. Finally, the points are drawn, while maintaining a floating horizon. This procedure is also discussed separately.

4.6.4 Line Drawing Order

This line drawing order is based on the viewer position, and consists of two loops. The matrix axis (X or Y) along which the inner loop (see Table 5) proceeds is chosen such that the drawing order is as near the screen horizontal axis as possible. The direction in which points are taken along the matrix axis is either near range to far range, or arbitrary if the axis is sufficiently close to horizontal (in which case adjacent points do

not vary substantially in range). For the outer loop (see Table 5) the progression is near range to far range, in accordance with the requirements of the floating horizon algorithm.

Look Direction		Inner Loop			Outer Loop		
#	Degree Range	Start	End	Inc	Start	End	Inc
0	337.5 - 22.5	Min X	Max X	+1	Min Y	Max Y	+1
1	22.5 - 67.5	Min Y	Max Y	+1	Min X	Max X	+1
2	67.5 - 112.5	Max Y	Min Y	-1	Min X	Max X	+1
3	112.5 - 157.5	Min X	Max X	+1	Max Y	Min Y	-1
4	157.5 - 202.5	Max X	Min X	-1	Max Y	Min Y	-1
5	202.5 - 247.5	Max Y	Min Y	-1	Max X	Min X	-1
6	247.5 - 292.5	Min Y	Max Y	+1	Max X	Min X	-1
7	292.5 - 337.5	Max X	Min X	-1	Min Y	Max Y	+1

Table 5. Line Drawing Order for Hidden Line Removal

4.6.5 Floating Horizon Algorithm

The basic assumption which defines the floating horizon algorithm is: for a given x, if the y value of the line is larger than the y value for any previous line, then the point is visible. If the y value is less, then the point is hidden. This assumption is true for a top surface. There are also provisions for viewing the under surface of an area, by tracking the current minimum y. This is not relevant to the Terrain Visualization project, and was therefore not implemented.

The algorithm makes the assumption that the y value is known for each x location. The only locations that are directly known are the transformed elevation post values. The values for each x can be computed through the use of a line-generation algorithm. The efficiency of the line-generation algorithm is therefore critical to the performance of the hidden mesh display. The line-generation algorithm previously discussed (section 4.5) will be used to implement this algorithm.

The algorithm also makes the assumption that the areas are drawn in planes from near to the viewer to far from the viewer. This is accomplished through the use of a strict, pre-defined surface processing order. This drawing-order algorithm is also discussed separately (section 4.6.4).

4.7 Continuous Surface Generation

Terrain shading, which includes the draping of scanned data, offers an outstanding method of conveying information to the analyst. There are a variety of information sources which might be presented as a perspective overlay:

* Digital Map Data - Increasingly, scanning equipment is utilized to digitally sample topographic mapsheets. Scanned data must be corrected through resampling to account for map projection, and mapsheet and scanning distortions. Cost effective, high resolution scanners (in terms of both sensitivity and spatial sampling rate) are available. Video-disk and CD-ROM devices offer other sources of sampled data for perspective draping.

* Terrain Analysis Products and Source Data - The AirLand Battlefield Environment (ALBE) program has demonstrated the utility of both mobility and intervisibility prediction overlays. Vector-based lineal data are used directly. Areal features may be handled either by rasterizing or by simulating with multiple internal vectors. Product and source-data overlay offer both strategic and tactical insight. Additionally, they afford a qualitative assessment of a predictions accuracy - permitting visual correlation of model output and the contributing terrain.

* Remotely Sensed Imagery - Commercial satellite (e.g. SPOT) imagery stands as a potential invaluable reference source. Due to its high spatial resolution and potential immediacy, this data form also holds promise as a terrain-analysis data source. The perspective draping of image data also facilitates comparative assessment of the accuracy of standard terrain analysis source data.

Digital-map and remotely sensed imagery generally contains inherent shading information which should not be subjected to calculations based on the local terrain. Terrain analysis products and source data, on the other hand, may be presented so that the intensity of the displayed data is varied according to the orientation of the terrain with respect to the selected sun angle.

This shading technique is demonstrated in Figure 13. Intensity is assigned based on the cosine of the angle between the surface normal and the illuminating direction. This shading model is referred to as the Lambertian or constant diffuse shading model. Lambertian shading requires the assignment of a constant global ambient illumination to avoid total darkness for faces which receive no direct illumination - lower right orientation in Figure 13.

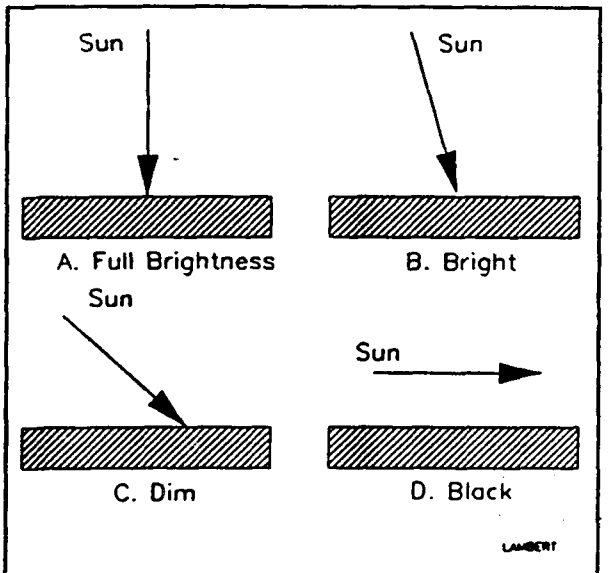


Figure 13. Lambertian Shading

Unrestricted constant shading requires a high resolution image buffer. By limiting the number of source or product colors and the number of intensity levels output by the Lambertian shading model, a look-up-table may be constructed to generate pseudo color images. An 8-bit image buffer may support up to 16 shades for 16 different product or source data categories.

A natural adjunct to shading models is the concept of smooth shading. Smooth shading techniques were first introduced for use in the display of curved surfaces. With Gouraud shading (the most dominant smooth shading technique), the polygons comprising the terrain surface are shaded such that intensity discontinuities (arising from changes in the surface normal) are eliminated across adjacent edges.

Gouraud shading assumes that the sampled surface in fact represents a smooth function. Surface normals at terrain sample point may be calculated by examining adjacent terrain elevation values. Sample normals are used to calculate intensity values at the sample points. After transforming and projecting the sample points onto the viewing surface, a set of 2D screen coordinates and associated intensity values are known. To determine the shading points inside the polygonal area, two successive linear interpolations are performed.

Within the progressive refinement pipeline, the Gouraud shading technique is applied inconspicuously. Utilizing the same front-to-back Z-buffering technique utilized within the Hidden Surface Removal Generation element, the effect is a natural smoothing of intensity between adjacent polygonal areas of similar attributes. Figure 14 illustrates a Gouraud shaded terrain image.

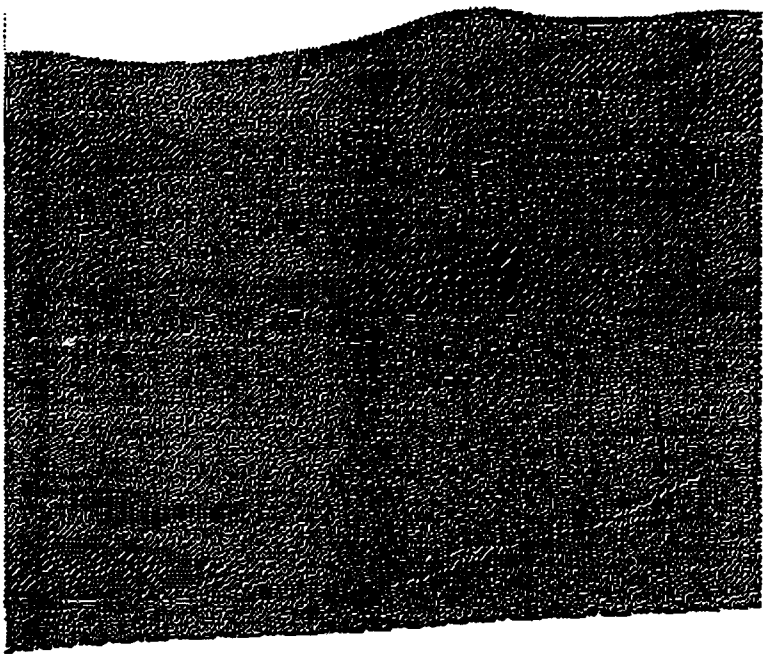


Figure 14. Perspective Projection with Even Spacing

4.7.1 Description

This step in the image refinement process transitions from a hidden-line mesh perspective to a continuous-surface perspective view. Two hidden-surface algorithms commonly used in terrain rendering are the Z-buffer and back-to-front painter's algorithm. Unfortunately, both of these algorithms generally rely on the overwriting of previously drawn data to achieve the final image. However, the Z-buffer algorithm can be supplied data from front to back such that data overwrite does not occur. The resulting visual effect appears as the unrolling of a colored carpet of terrain detail extending from the viewer and undulating with the terrain mesh as it proceeds to the visible horizon.

This capability represents a facility of demonstrated value this is a candidate software upgrade in the future P3I program for DTSS. The resulting software can be transported to the DTSS and become a standard terrain analysis product.

4.7.2 Inputs and Outputs

The parametric inputs necessary to generate the continuous surface consist of viewer position, sunshading, elevation exaggeration and surface overlay type specifications. These are provided during the General and Interactive Parametric Input steps.

DTED data, and the specified overlay data are also necessary inputs. For the latter, the overlay material can consist of digital map data [section 4.2.1.1], satellite imagery [section 4.2.1.2] and terrain analysis data [see the DTSS DTDB design document].

The output of this step in the progressive refinement chain is the continuous surface display.

4.7.3 Processing Steps

The generation of the continuous surface begins with the transformed [screen] coordinates of the input elevation data array. Each point is considered to be the corner of a quadrilateral surface element whose boundaries are determined by its neighbors (as an example, for a northward looking view, the neighbors are the points lying immediately to the north, east and northeast of the point being processed).

The quadrilateral is then divided into two triangular facets. This process segments the data sets such that three points in screen coordinates can be directly related to their original geographic coordinates.

The next step is to develop the transformation coefficents relating screen coordinates to geographics for the three vertices of the triangle. With three coordinates, a six parameter transformation equation can be determined which includes the effects of (1) X/Y Translation, (2) Perspective X/Y Scaling and (3) X/Y Axis Rotation about the look point.

Following this, the transformation coefficients are used to fill the triangles with overlay material. The triangle is scan line filled, with linear interpolation parameters computed for the start and end of each line [using the transformation equations computed for the vertex positions]. As each screen pixel is processed, the corresponding overlay array pixel's map indices are computed using the interpolation parameters and the array's intensity/color value inserted into the screen buffer at the screen pixel location.

4.7.4 Pseudotiling Algorithm

A step which significantly accellerates the processing of input elevation array data for perspective transformation purposes is the division of the array into 16 x 16 subarrays ("pseudotiles"). The pseudotiles allow the computation of initial window intersections to be performed on a tile basis rather than a point basis, considerably reducing processing requirements. In addition, sampling calculations [to reduce processing requirements at far range] are conveniently made on a tile by tile basis. Tiling, as applied to terrain visualization, has been applied to be specific to perspective transformation accelleration.

It would be possible to extend the tiling concept to other terrain analysis applications, such as intervisibility products. In this case, tiling would be used to speed data access by reducing data volume. Restructuring Intervisibility code (to allow all lines of sight falling within a tile to be simultaneously processed) would further reduce data I/O and therefore

speed processing. For this application it might be desirable to convert the virtual tiling scheme to actual physical tiling to obtain maximum performance.

The tiling process is illustrated in figure 15 and described below.

Starting with the observers location, the elevation matrix is segmented initially by defining tiles which are each 16 x 16 elements in size (see the upper portion of figure 15). Each tile is then augmented by adding to it one row and one element per row in each direction (so that each tile will have an overlap of one with its neighboring tiles, thus providing a seamless elevation array structure).

With this uniform tiling of the elevation array, the processing is constructed such that initially each tile is processed as an entity rather than by individual elevation elements.

For each tile, the intersection of the tile with the view window is computed. If no intersection exists, the tile is discarded. This prevents processing of the elements within the tile. If a tile intersects the window, the position of the tile relative to the observer is used to compute elevation interpolation-interval parameters [if an enhanced surface is to be generated with spline or fractal interpolation]. Again this prevents interpolation intervals from being computed on an element-by-element basis.

The tiles are numbered beginning with zero. The zeroth tile is the tile immediately to the northwest of the observers position (see figure 14). They are numbered in a clockwise manner. As the tile numbers increase, range increases as well, in the manner depicted in figure 15.

By processing the tiles in numerical order, data closest to the observer will be processed first. This characteristic provides feedback to the operator immediately concerning information of the greatest importance to the product being generated.

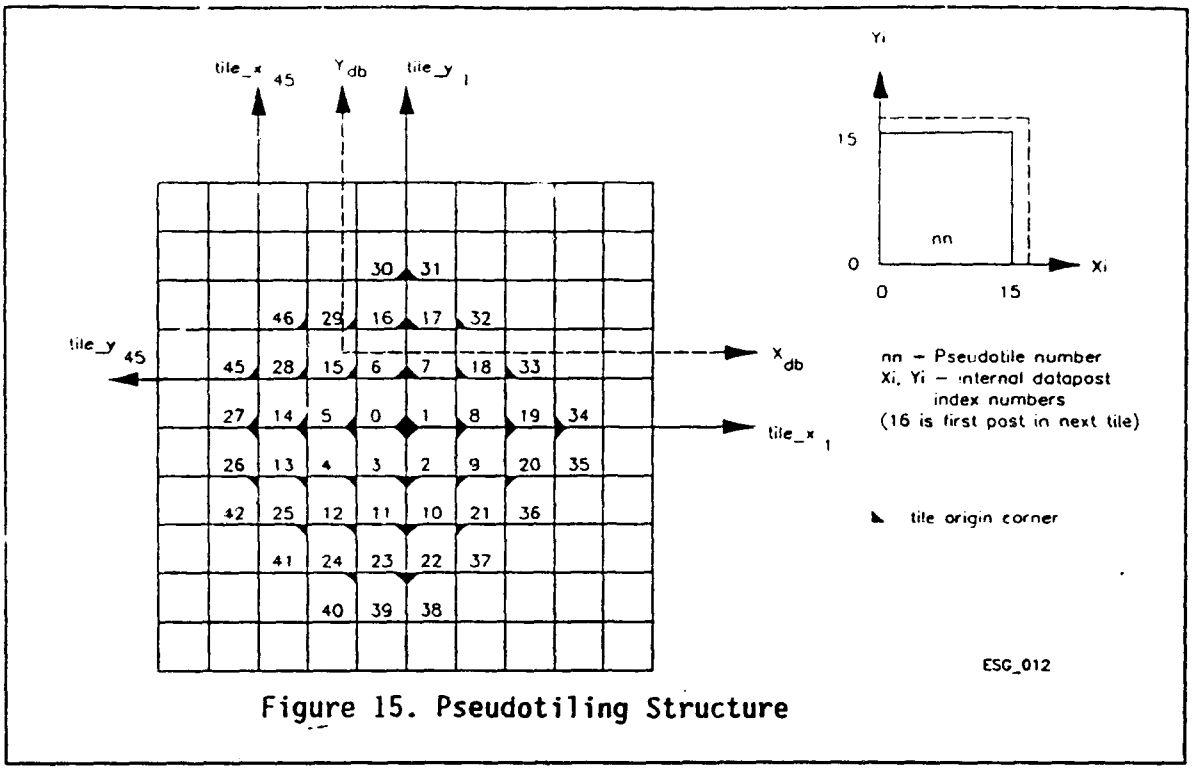


Figure 15. Pseudotiling Structure

4.8 Enhanced Surface Generation

Three methods of extending terrain visualization to greater realism are described in this section. These methods are mutually exclusive within the progressive refinement pipeline. They are described together due to their common goal of a more accurate visual depiction of the terrain surface.

Polygon techniques owe their popularity to the fact that they are rough approximations. Polygon processing techniques are easily understood and hardware implementations of polygonal scan-line algorithms are well established and readily available. The techniques described herein seek levels of visualization requiring a departure from traditional polygonal techniques.

4.8.1 Adaptive Forward Differencing

Adaptive forward differencing is an incremental evaluation technique which extends forward differencing by allowing for the dynamic adjustment of the incremental variable. For the Terrain Visualization program, three levels of incrementing were implemented: 1.) subdivision, 2.) database sampling, and 3.) sparse sampling.

A simplified explanation of the generation of a new surface is as follows. For a given data base cell, first compute the distance from the viewer, and then from this compute how many screen pixels this cell will occupy. If the cell occupies much more than some pre-defined (by operator) threshold (relating to ground meter spacing), then this cell is a candidate for subdivision (subdivision techniques will be discussed further in the next sections). This subdivision occurs at some increment such that the number

of pixels that the smaller cells will occupy is the threshold which was defined by the operator. The new smaller cells are defined by new elevation posts which are spaced uniformly within the old cell. The color associated with the new cell is determined from either imagery, feature data, or shaded elevation, just as the original was. The new cell is then drawn, just as the original: transform to 2 dimensional screen coordinates, then fill the cell.

Conversely, if the cell occupies much less than the pre-defined threshold, then this area is a candidate for sparse sampling, where original database points are simply dropped from the calculations, in order to eliminate unnecessary processing.

In order to further explain the effect of the forward differencing, the following figures are discussed. Figure 16 is an illustration of the original database spacing from a perspective viewpoint. Figure 17 shows the same spacing, but with a viewpoint of directly above the area.

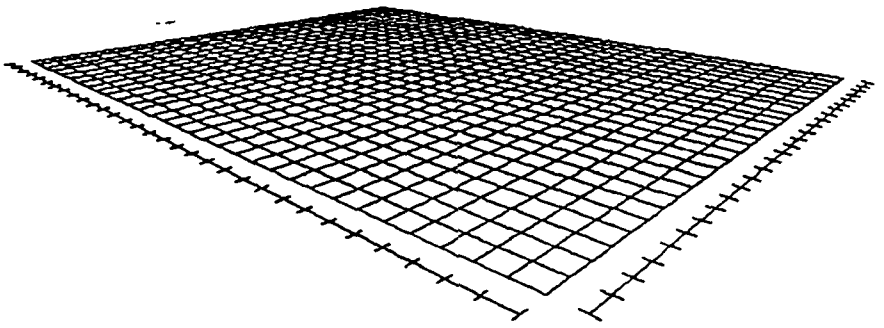


Figure 16. Perspective Projection with Even Spacing

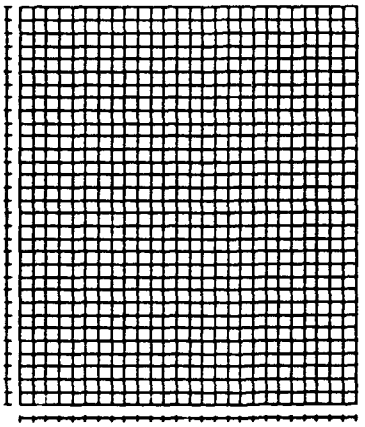


Figure 17. Vertical Projection with Even Spacing

Next, in Figure 18, the same imaginary area is then shown in a perspective view, but with sparse sampling introduced as the distance from the viewer is increased and/or with subdivision of the original grid done in the regions near the viewer. Note that the screen cell size remains relatively constant, in contrast to Figure 16, where the cell size becomes extremely small. The effect of the database sampling is illustrated in Figure 19, where the same area is now shown once again from the viewpoint directly above the area. The areas which are nearer to the viewer (lower left) cover a much smaller database area, than areas far from the viewer (upper right).

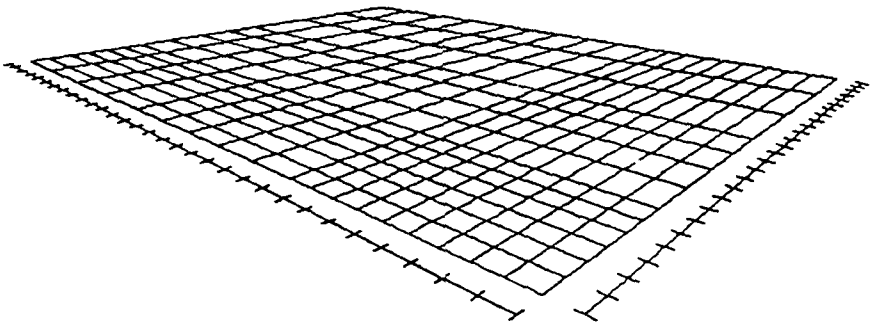


Figure 18. Perspective Projection with Adaptive Grid

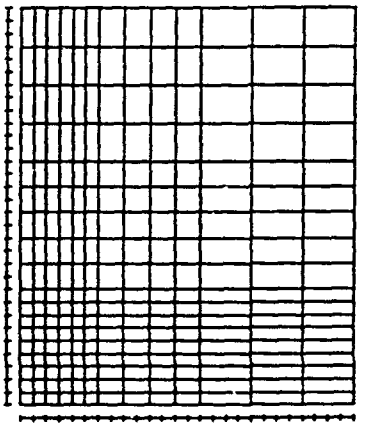


Figure 19. Vertical Projection with Adaptive Grid

The calculations used for the sampling are explained in the following paragraphs. Sparse sampling begins when the spacing in the x direction (x_steps) ≥ 2.0. Interpolation begins when x_steps < 1.0. The same is also true for sampling in the y direction (y_steps).

First calculate x_steps_R and y_steps_R in the reference database plane:

$$x\_steps_R = \left(\frac{SL_o}{f_o}\right)\left(\frac{\sin\Theta_o * \cos(\Theta - \Theta_o)}{\sin\Theta}\right)\Delta x_s$$

Δx_s = desired horizontal pixel spacing on the screen

Θ = depression angle to actual database point where x_steps_R is being determined

f_0 = observer to view window perpendicular distance

SL_0 = slant range from observer (viewpoint) to the origin of the reference plane

$\Theta 0$ = depression angle of the LOS (line along which SL_0 is measured) relative the horizontal

Similarly:

$$y\_steps_R = \left(\frac{SL_o}{f_o}\right)\left(\frac{\sin\Theta_o * \cos(\Theta - \Theta_o)}{\sin\Theta}\right)\left(\frac{\cos\alpha_{max}}{\sin(\Theta + \alpha_{max})}\right)\Delta y_s$$

α_{max} = maximum expected terrain - slope angle (precomputed, not a user input)

Δy_s = desired vertical pixel spacing on the screen (computed from user input specification of desired ground spacing)

Note: x_steps_R and y_steps_R are in length units.

Next, x_steps and y_steps in the non-rotated reference plane are calculated (see Figure 20.) where X and Y are aligned with the database grid (Y is usually towards the north).

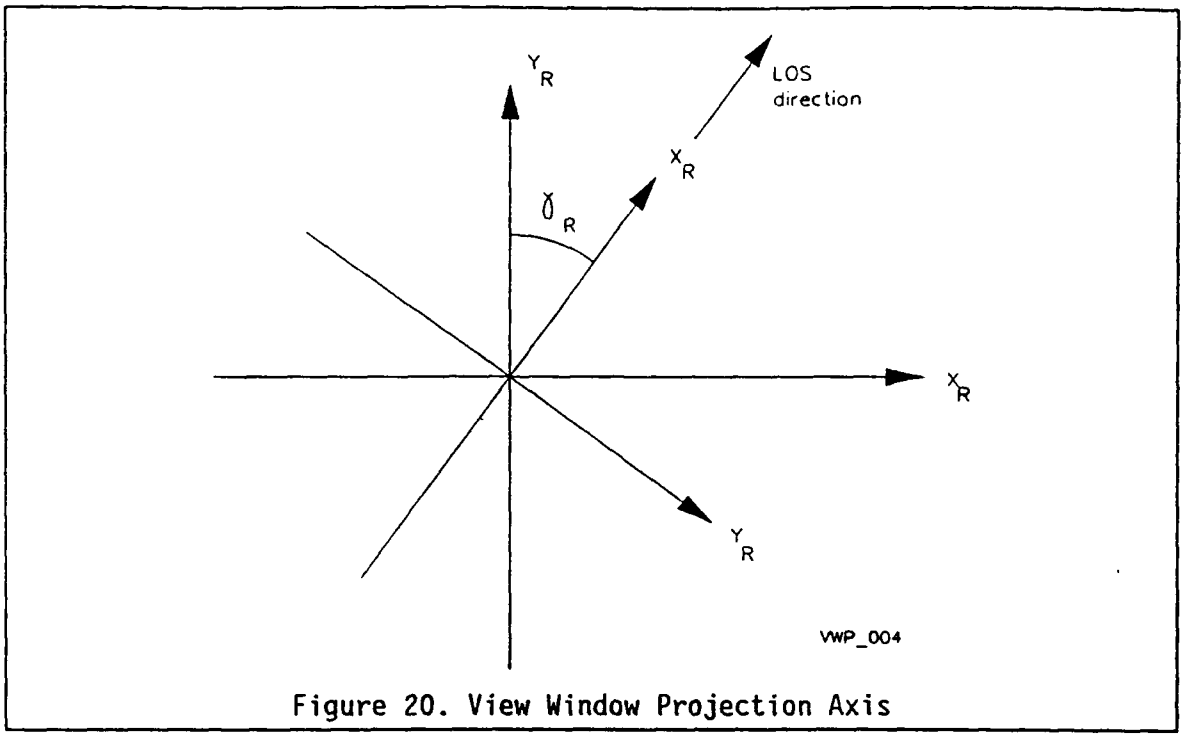


Figure 20. View Window Projection Axis

$$x\_steps = \left(\frac{1}{\Delta X_{DB}}\right)\left(\text{MIN}\left[\frac{x\_steps_R}{|\cos\gamma_R|}, \frac{y\_steps_R}{|\sin\gamma_R|}\right]\right)$$

$$y\_steps = \left(\frac{1}{\Delta Y_{DB}}\right)\left(\text{MIN}\left[\frac{y\_steps_R}{|\cos\gamma_R|}, \frac{x\_steps_R}{|\sin\gamma_R|}\right]\right)$$

γ_R azimuth angle of the LOS

ΔX_{DB} database grid spacing

ΔY_{DB} database grid spacing

where ΔX_{DB} and ΔY_{DB} are in the same length units as f_0, SL_0, ΔX_s, and ΔY_s.

4.8.2 Tensor Based Surface Generation

As described earlier, the rendering or generation of images from gridded samplings is generally accomplished through the shading of polygonal elements formed from adjacent sample points. Even the most sophisticated continuous shading models can result in visually objectionable artifacts from the data representation technique. Mach banding is apparent at the borders between adjacent polygons with differing intensity gradients. Also, with polygonal techniques, there remains an unnatural angularity to polygonal silhouettes against horizon edges within the terrain.

Less understood and not as well established are free-form surfaces. "Free-form" surfaces, as the name implies, embody a substantial class of mathematical descriptions which do not have the same sort of regularity that makes polygonal techniques so convenient. Of most importance to the present discussion, is a subclass of free-form surfaces constructed from tensor products. A tensor product surface is formed from:

$$S(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} p_{i,j} F_i(u) G_j(v)$$

Where: p_{ij} are control points
$F(u)$ and $G(U)$ are "blending" functions

A number of image surface generating tensor product bases have been investigated for computer graphics modeling. These include the Bezier basis, Hermite basis, B-Spline basis, Catmull-Rom basis and Beta-Spline basis. Given that the gridded terrain data represents accurate surface elevation values, interpolating rather then approximating bases are more appropriate.

Of particular interest to terrain visualization, the interpolative Catmull-Rom basis offered the capability to select a surface "tension". The tension parameter causes the surface to bend more sharply by increasing the magnitude of the tangent at the control points. Catmull-Rom curves also exhibit the desirable properties of affine transformation invariance, global-smoothness, and local control. It is through the use of this algorithm that the surfaces are subdivided.

Within the progressive refinement pipeline, surface rendering is applied inconspicuously. Utilizing the same front-to-back Z-buffering technique to drive adaptive forward differencing, the effect is a natural smoothing of shape and intensity of the terrain surface.

In order to further illustrate the enhanced surface generation process, the following figures are included. First, in Figure 21, the original elevation post spacing is shown with a hypothetical surface draped over the points. Next, in Figure 22, one of the cells is illustrated with newly generated interpolated points. Finally, in Figure 23, the delta x and y offset coordinate system (u and v coordinates) is shown.

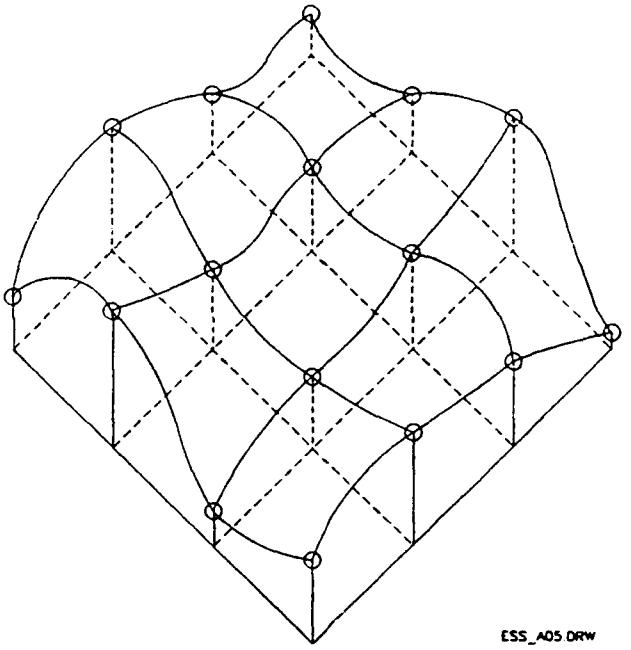


Figure 21. Original Elevation Post Spacing

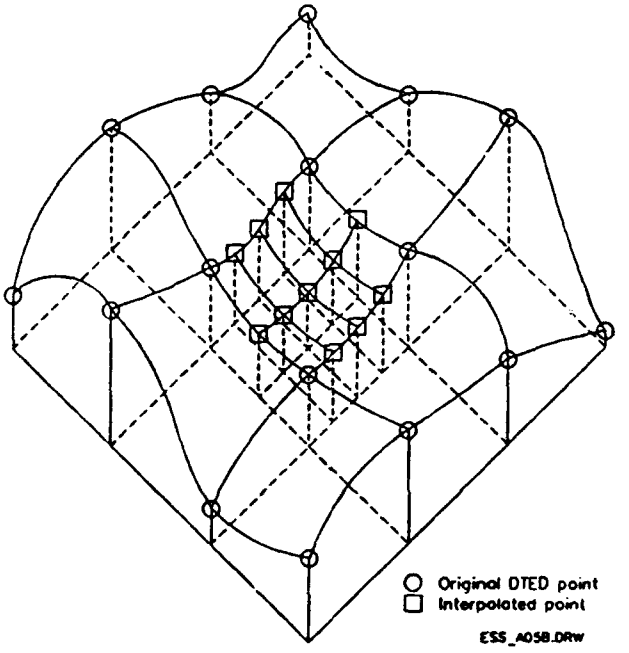


Figure 22. 2:1 and 4:1 Interpolated Data Posts

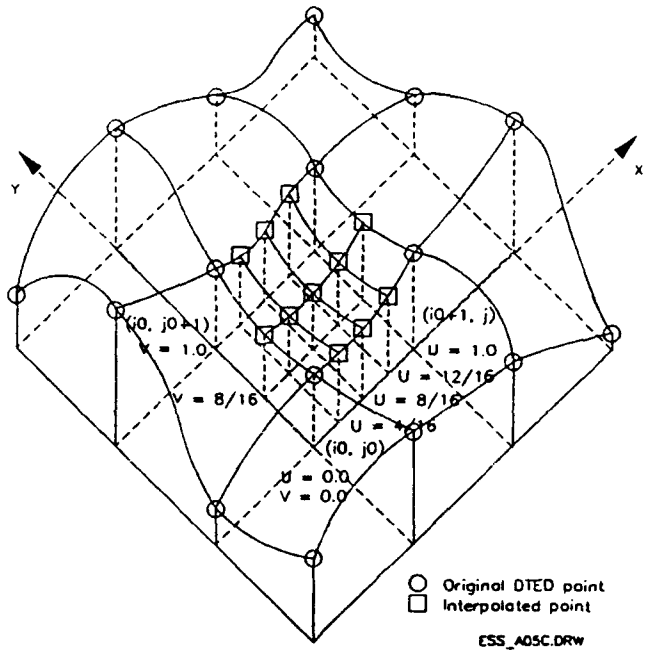


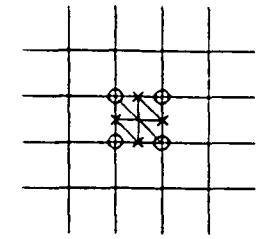
Figure 23. Offset (u,v) Coordinate System

4.8.3 Fractal Surface Generation

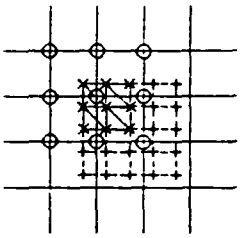
Fractal methods for terrain visualization have been used to allow data base amplification - the controlled addition of random detail as well as interpolation of a sparse terrain description. Fractal interpolation methods are compared in Figure 24. Early fractal interpolation methods (Triangle Edge subdivision, Figure 24 and Diamond - Square subdivision, Figures 24 and 25) generated visual defects along specific boundaries. A method described by Miller in [8] as square-square subdivision (Figures 24 and 26), does not exhibit this "creasing" defect and offers very realistic terrain visualizations. However, it has the problem of not replicating the original dataposts as part of the interpolated database. The diamond-square method does not have this latter problem, but does have second-order surface discontinuities. Because of an expressed concern by USAETL about not including the original datapost values in the interpolated database, the diamond-square method (as well as the square-square method) has been implemented.

Primary concerns of PAWS-based fractal surface generation are image generation time and subdivided data volume. By carefully controlling the amount of displacement which may occur, major portions of hidden areas may be bypassed by the subdivision process. Subdivided data volume may be reduced by incremental subdivision where only those portions of the image which are being rendered are subdivided.

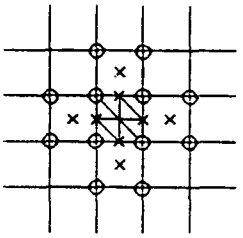
Within the progressive refinement pipeline, fractal subdivision is applied inconspicuously. Utilizing the same front-to-back Z-buffering technique, the effect will be a gradual detailing of surface shape.



Triangle–edge Fractal Subdivision



Square–square Fractal Subdivision



Diamond–square Fractal Subdivision

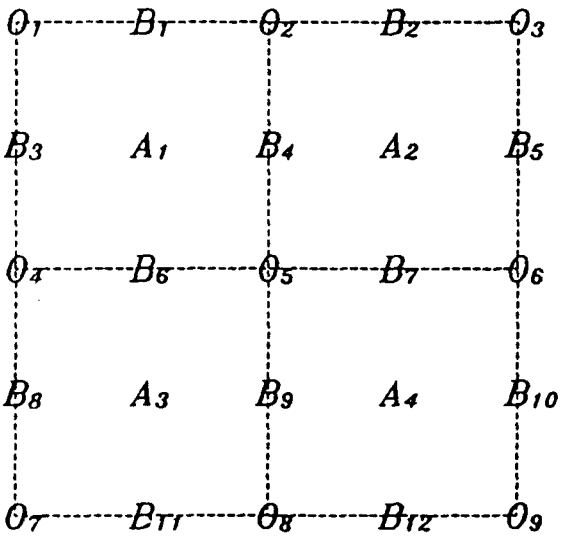
○ Original dataposts used in defining the interpolated cell

× Interpolated datapost locations

\+ Interpolated datapost locations

ESG_016

Figure 24. Comparison of Fractal Subdivision Methods



$$A_1 = \frac{1}{4}(O_1 + O_2 + O_4 + O_5)$$

$$A_2 = \frac{1}{4}(O_2 + O_3 + O_5 + O_6)$$

$$B_4 = \frac{1}{4}(A_1 + O_2 + A_2 + O_5)$$

$$B_6 = \frac{1}{4}(O_4 + A_1 + O_5 + A_3)$$

ESG_18

Figure 25. Diamond-Square Fractal Subdivision

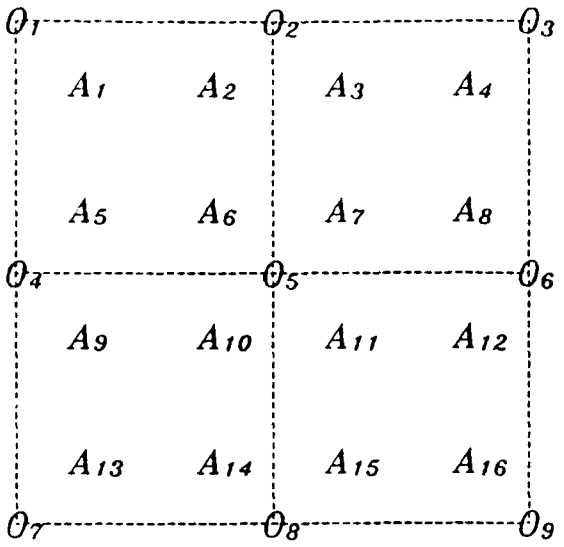


Figure 26. Square-Square Fractal Subdivision

4.9 ITD Feature Overlay Generation.

To further image realism and add to the visualization capabilities of the terrain analyst, specific items from the ITD thematic data and operator parametric input data may be displayed. The items include trees and tree lines (deciduous and conifer), streams (lines of drainage), surface roadways, and buildings. Items are represented geometrically in perspective scale and are subject to hidden surface elimination. Items appear according to applicable attributes recorded in either the ITD thematic data or operator specified parametric data. Additional data base items may be added according to relative value to terrain analysis and completion of other areas of the progressive refinement process.

4.9.1 Feature Overlay Types

4.9.1.1 Trees.

The recent technological past has seen the proposal of many new tree representation and synthesis techniques. In these works, two classes of tree generation have emerged. The first class is characterized by topological methods which generate near-perfect tree images. The second class is characterized by development models which allow for growth strategies and introduce stochastic and recursive growth patterns. Very realistic images have been generated.

Visible details of rendered trees should vary according to the perspective distance of the tree. A hierarchy of tree-rendering techniques is proposed to avoid the generation of unvisualizable detail:

* Detailed tree generation - Tree imagery through combinatorial analysis for distances from 0 to 350 meters.

* Nominal tree generation - Tree generation through a combination of the detailed tree generation and coarse tree generation techniques weighted by distance for trees at distances from 350 meters to 1050 meters.

* Coarse tree generation - Tree imagery through stochastic sampling of a 2D texture map for distances greater than 1050 meters. Generally, only side views of trees are visible at this distance and, while the silhouette and trunk may be partially discernable, individual leaves and branches are indiscernible.

Data base attributes which will influence tree appearance are the tree type (deciduous and/or conifer), season (deciduous leave coverage), stem diameter, and tree height.

4.9.1.2 Surface Roadways.

Roads and their attributes are derived from the transportation information within the ITD data. All of the attributes specified within the ITD will be supported. This includes road structure category, surface type, weather type, travelway characteristics, existence classification, accuracy, overlay, slope or gradient, and width. Items other than width will be indicated by the color of the line.

4.9.1.3 Rivers and Drainage

These are displayed in the same manner as are roads. In addition to width (which is displayed geometrically), characteristics such as depth, flow, intermittancy, etc. are indicated by color coding.

4.9.1.4 Buildings.

Individual building definitions will be supported through operator parametric specifications. Both cylindrical and parallel piped shaped structures may be specified. Supported roof types include gabled (traditional), gambrel, and flat (no roof). Building location, height, width and breadth, and orientation will also be selectable. Complex building shapes may be constructed by composite definitions.

The rendering of buildings will consist of forming planer components of the exterior surface, computing shading values from the surface normal, transforming the surface to the viewing reference frame, and subjecting the projected coordinates to Z-buffer hidden surface elimination on the existing image.

The implementation of ITD data overlays has been implemented by providing the software to display vector line overlays (roads and streams - could be utilized for any type of color-coded line track to be draped over the terrain), individual models (specified tree and building point models) and vector lines of specified or randomly selected point models (tree lines, etc.) from an input file which has been created by extracting ITD data from the DTSS database (from the appropriate ARC/INFO files).

5 Summary and References.

This section summarizes the proposed terrain visualization research.

5.1 Program Summary.

The Terrain Visualization program described above implements a progressive refinement technique. The technique offers a compromise between the demands of interactivity and realism.

The program proposed a PAWS-based implementation for rapid transition to an end product for the DTSS. The effective image generation rates achieved by the DTSS prototype perspective plot model was improved nearly 100 fold to become interactive.

A goal at the beginning of the program was the evaluation of alternative interpolation techniques in providing enhanced terrain detail to the operator. The insight gained from initial use of the package is detailed below:

1. Flat Shading: This is the simplest interpolation option [it corresponds to simple linear interpolation]. It results in the fastest display since a single slope value is computed from neighboring elevation points and used to fill the polygonal region so bounded.

Speed is the principal advantage of flat shading. It also directly displays the original DTED points since these are used as the polygon boundaries. Thus it might be used to inspect the DTED values.

The flat shaded display is not inherently a more accurate display. The earth's surface is highly continuous [the elevation value for Mount Everest is not adjacent to sea level, for example]. The simple linear interpolation assumed by flat shading possesses sharp discontinuities at the boundaries between polygons which do not occur in the real world. These discontinuities are the reason the original DTED values are visible. In addition, natural terrain features, such as hills, are generally composed of smoothly varying surfaces rather than the faceted appearance provided by flat shading.

2. Gouraud Shading: As described in section 4.7, Gouraud shading interpolates intensity values derived from vertex surface normal calculations. Gouraud shading possesses zeroth order continuity [i.e., value continuity]. It provides a visual impression of smoothly varying terrain slope within the original (DTED provided) polygon boundaries.

At the boundaries between polygons, Gouraud shading does not possess first order continuity. This can give rise to mach banding defects in the image which are not present in the real-world.

In addition, since Gouraud shading interpolates intensities rather than normals, highlights on the surface will not generally be present unless they are spread over a large area.

Although the calculations required for Gouraud shading are somewhat less than more complex interpolation procedures such as the spline, this reduction is largely masked by the requirement to calculate Z- buffer values, calculate and fill the interpolated intensity values in the buffer and data

access. The principal advantages of Gouraud shading are, therefore, value continuity (as compared to flat shading) and the lack of terrain facets (as compared to flat shading and spline interpolation). It renders a terrain view which is more realistic than flat shading.

3. Spline Interpolation: The spline surface possesses both zeroth and first order (slope) continuity. Interpolated elevation values lie on this surface. The spline therefore allows surface normals to vary in a more realistic fashion than does either flat shading or Gouraud shading.

The degree of interpolation provided by the spline is a function of the terrain resolution (in meters) specified by the operator. A small specification will generate more values along the spline than will larger values. For large values, scenes produced by the spline will be quite similar to flat shaded scenes.

The lack of continuity defects, as compared to flat and Gouraud shading, allows spline interpolation to produce values which vary in a more realistic fashion. Spline interpolation is therefore considered superior to flat or Gouraud shading with regard to providing a realistic surface. Since spline interpolation uses flat shading, the terrain will have a faceted appearance which, if the spacing is large, may be visually distracting.

4. Fractal Interpolation: As noted previously, the fractal interpolation method possesses zero and first order continuity with respect to the original terrain points. It is, however, built as an iterative process which successively uses the last terrain values produced to generate a new set of intermediate points. Various linear or quadratic interpolation methods are used in this process as a function of the interpolation type (see section 4.8.3). This methodology produced terrain with a terraced appearance. No significant improvement in speed for fractal interpolation was noted.

As a result of experience gained in using the Terrain Visualization software with these four interpolation methods, it is felt that the two most logical methods for utilization are flat shading (for speed) and spline interpolation (for visual accuracy). The smoothness of the polygonal intensity variation provided by Gouraud shading is not considered to provide significant benefits to outweigh the realism of the spline function. Fractal interpolation provided no significant speed improvement to compensate for the terraced appearance of the terrain.

5.2 Recommendations for System Modifications

5.2.1 Integration Into DTSS

The Terrain Visualization software package has been implemented via a development environment which focused on demonstration of the technical feasability of using the DTSS PAWS as an effective visualization tool. As a consequence, the following observations concerning the TV software package are appropriate relative to its' potential utilization as an integral DTSS function:

1). Application workspace utilization: The Terrain Visualization software currently assumes that a single workspace exists which is used for loading and processing. This is distinctly different than DTSS, which partitions workspaces between loaders, the master data base and user workspaces.

2). MMI Utilization: The standard DTSS Man-Machine Interface is used for the TV applications software. This substantially contributes to easing the integration of the software into DTSS.

3). Driver Compatability: The TV software bypasses the NOVAGKS package used by DTSS and accesses the Parallax driver directly. This is not a problem for the generation of product dislays, since only one product type (standard DTSS versus Visualization) is displayed at one time. However, it does suggest that modifications of the basic procedures used for final product review and hardcopy generation would need to be implemented in order to integrate TV capability into the DTSS environment.

4). Data Sizing [1:50,000 scale]: The following estimates for a 15 minute x 15 minute map are are provided for utilization of the TV software:

Raster Map Data [8 Bit/pixel]: 36 Megabytes
DTED Level I: 200 Kilobytes
Spot Imagery [One 10 Meter Layer]: 9 Megabytes
Landsat [Single 30 Meter Band]: 1 Megabyte
Formatted ITD: 5 Megabytes

Thus a total of 51 megabytes of storage is required for the range of sources for TV use in a Tactical planning application. This number can vary as a function of source availability and the desire to support, for example, multiple bands of multi-spectral data.

5. Data Sizing [1:250,000]: The following estimate for a 1 degree x 1 degree map area is provided [note that at this scale paper maps are normally 1 x 2 degrees, however tiles within the TV software are not larger than 1 degrees squares, reducing data sizing and coverage]:

Raster Map Data [8 bit/pixel]: 24 Megabytes
DTED Level I: 3 Megabytes
Spot Imagery [One 20 Meter Layer]: 36 Megabytes
Landsat [Single 30 Meter Band]: 16 Megabytes
Formatted ITD; 25 Megabytes

Thus a total of approximately 104 megabytes of storage is required to support a range of product displays at the Planning level of utilization.

Suggested Implementation

The following general strategy is suggested for implementation of Terrain Visualization capability on DTSS.

1) Data Loading

The current DTSS [DTED and ITD] loader software is executed as a separate process which transfers data into an intermediate data loader workspace. Specific [Data Base Utility CSCI] software then formats and transfers data into the master DTDB workspace. Area specific workspaces may then be populated.

For Spot and Landsat data, it is suggested that the TV developed loaders are logical candidates for future DTSS utilities dealing with Multi-Spectral data. Assuming the addition of tape loading capability to DTSS, The operating scenario for this utilization is:

A). The Spot and Landsat loaders will load operator selected tape resident image files into the intermediate loader workspace.

B). The display of Spot and Landsat data and subsequent computation of affine transform coefficients will be an interactive utility which acts as a precursor to extraction and transfer of the data into the master DTDB data base. Existing Master Data Base [MDB] pointer and index files will be modified to permit MDB utilities to recognize Spot and Landsat data.

C). Once in the MDB, utilities will be formulated to populate specific workspaces with SPOT and Landsat information.

For ITD data, workspace population will use master data base formatted data. Specific extraction utilities [derived from those developed under the current TV effort] will provide the necessary capability to format resident ITD files directly for the generation of TV displays.

For DTED data, existing TV loader software must be modified to use DTSS DTED loader data. It will formulate the necessary DTED tiles directly from master data base resident information and populate workspaces.

2). Product Generation

The existing progressive refinement chain will be integrated as a standard product generation process under control of the DTSS system supervisor utility. It will replace the existing [wireframe] perspective view capability. It will be formulated as a series of interactive processes. The last step in the chain will generate both a display and a stored [raster] image of the final product.

3) Final Product Review

The existing DTSS final product review function will be modified to allow display of the raster bit map of the visualization product. This will allow visualization products to be annotated for further utilization in planning.

4). Hardcopy Output

Current DTSS hardcopy devices are not adequate to provide hardcopies of Terrain Visualization images. An imaging printer [or an electronic link to

such a device] would provide this capability.

Other Considerations

The majority of the TV software has been developed in the MIL-STD-1815A Ada programming language, with the sole exception of the graphic device calls which are written in C. DOD-STD-2167 was not imposed as a program requirement, nor was it conformed to during the TV development effort.

Since the software implementation has largely been completed, the migration to DOD-STD-2167 should be limited to only life cycle maintenance considerations. The generation of a Data Base Design document and Software Design document would provide the information necessary to maintain the TV software. Efforts expended in Software and Interface Requirements Specifications and testing documentation would provide only a relatively small return for the effort invested.

The software would ultimately be placed under DTSS configuration management.

5.2.2 Migration of Software to New Systems

While the primary beneficiary of the program would be the DTSS, the program has potential benefits which transcend the DTSS mission. The benefits of a real-time terrain image generation tool have been demonstrated by the CIG program. This program seeks to move some of those benefits to general purpose platforms. The Army-wide standard of future perspective view implementations will be elevated by the success of the Terrain-Visualization program.

5.2.3 More Models

The proposed effort includes perspective visualizations of attributes associated with ITD data. This represents a new capability. While only a few ITD items are being supported, it is foreseen that this may open up new studies in methods to visualize attributed data.

5.2.4 Surface Draping Modification

Allow operator to modify surface draping source specification without redoing complete progressive refinement chain (operator must now specify sources to be used and their sequence of use in advance).

5.2.5 Additional Roughness

The interpolation method produces screen triangle sizes which are consistent with the desired generation - speed requirement. This method produces a smooth, continuous surface which sometimes may not be realistic in appearance. It may be desirable to introduce random variations in one or more of the following: 1.) the interpolated z values, 2.) the original DTED mesh points, and 3.) intensity / coloration of the triangular facets in the displayed image. This would introduce more roughness to the scene, which would increase the apparent realism of the display. This would of course, not increase the actual realism of the scene.

6 Acronyms

ALBE	AirLand Battlefield Environment
AGL	Above Ground Level
CIG	Computer Image Generation
DMA	Defense Mapping Agency
DTED	Digital Terrain Elevation Data
DTSS	Digital Topographic Support System
FSED	Full Scale Engineering Development
GFE	Government Furnished Equipment
ITD	Interim Terrain Data
LDS-A	Loral Defense Systems - Akron
LUT	Look-Up Table
MIPS	Million Instructions Per Second
MMI	Man Machine Interface
PAWS	Portable All Source Analysis System Workstation
PEO-CCS	Program Executive Office Command and Control Systems
RMS	Root Mean Square
TTD	Tactical Terrain Data
USAETL	U.S. Army Engineer Topographic Laboratories